Decoding Algebraic Codes

Heydtmann, Agnes Eileen

Publication date:
2001

Document Version
Publisher's PDF, also known as Version of record

Decoding Algebraic Codes

Agnes Eileen Heydtmann

Decoding Algebraic Codes

Decoding Algebraic Codes

Agnes Eileen Heydtmann

Ph.D. thesis

23. februar 2001

SUMMARY

This thesis focuses on decoding algorithms for different types of algebraic codes.

Two well-known algorithms for decoding alternant codes are compared — the Berlekamp-Massey and the Euclidean algorithm. It is proven that they are essentially equivalent in that all elementary calculations performed are the same, the intermediate results correspond directly to each other, and the sequence of steps in one can be traced as a sequence of steps in the other. This implies that the Euclidean algorithm for decoding alternant codes is as efficient as the Berlekamp-Massey algorithm.

Further, new decoding algorithms are developed for Reed-Muller codes and generalized geometric Goppa codes. Both are families of codes related to alternant codes. They have in common that codewords can be represented as the evaluation of certain functions.

In the case of Reed-Muller codes, Sudan's algorithm for decoding Reed-Solomon codes beyond half the minimum distance was adapted to the interpolation and factorization of boolean polynomials. Under the assumption that the weights of Reed-Muller codes are approximately binomially distributed, the algorithm decodes more errors than half the minimum distance as well.

In order to develop a decoding algorithm for generalized geometric Goppa codes that resembles the conventional algorithm for decoding geometric Goppa codes, it was necessary to study this generalization intensively. Once an appropriate inner product is defined, the dual of a given generalized geometric Goppa code can be described, and it is established that this code is a generalized geometric Goppa code itself. The knowledge about the dual code gives parity check equations which are made use of in the decoding algorithm.

Another decoding algorithm for generalized geometric Goppa codes is developed which is based on Sudan's improved algorithm for decoding geometric Goppa codes. Its error-correcting capacity is a considerable improvement in comparison to the first algorithm. Both algorithms can be used in decoding a promising construction of codes due to Xing, Niederreiter and Lam which concatenates generalized geometric Goppa codes.

RESUMÉ

Denne afhandling fokuserer på dekodningsalgoritmer af forskellige typer af algebraiske koder.

To velkendte algoritmer, Berlekamp-Massey algoritmen og den Euklidske algoritme, til at dekode alternante koder bliver sammenlignet. Det vises at de i det væsentlige er ækvivalente siden alle udførte elementære beregninger er de samme, mellemresultaterne svarer direkte til hinanden og følgerne af trin in den ene kan spores som følger af trin in den anden. Dette implicerer at den Euklidske algoritme til at dekode alternante koder er lige så efficient som Berlekamp-Massey algoritmen.

Endvidere udvikles nye dekodningsalgoritmer til Reed-Muller koder og generaliserede geometriske Goppa koder. Begge slags koder er i familie med alternante koder. De har tilfælles at kodeordene kan repræsenteres som evalueringer af visse funktioner.

For Reed-Muller koder er det Sudans algoritme til at dekode Reed-Solomon koder ud over den halve minimumsafstand som bliver tilpasset til interpolation og faktorisering af boolske polynomier. Hvis man antager at vægterne af Reed-Muller koder er tilnærmelsesvis binomial fordelt, så dekoder algoritmen også mere end den halve minimumsafstand af fejl.

For at udvikle en dekodningsalgoritme til generaliserede geometriske Goppa koder, som ligner den konventionelle algoritme til at dekode geometriske Goppa koder, var det nødvendig at studere generaliseringen intensivt. Når først et passende indre produkt er defineret, så kan dualkoden af en given generaliseret geometrisk Goppa kode beskrives og det bevises at denne kode selv er en generaliseret geometrisk Goppa kode. Kendskabet til den duale kode giver paritetstjekligninger som benyttes i dekodningsalgoritmen.

En anden dekodningsalgoritme for generaliserede geometriske Goppa koder udvikles som er baseret på Sudans forbedrede algoritme til at dekode geometriske Goppa koder. Fejlretningskapaciteten for denne er en betragtlig forbedring sammenlignet med den første algoritme. Begge algoritmer kan bruges til at dekode en lovende konstruktion af koder af Xing, Niederreiter og Lam som konkatinerer generaliserede geometriske Goppa koder.

ACKNOWLEDGMENTS

This thesis summarizes my research work at the Department of Mathematics of the Technical University of Denmark. It took place in the period between the 1st of February 1998 and the 31st of January 2001 under the supervision of Tom Høholdt and Jørn Møller Jensen. During the first year, I was supported by a graduate student scholarship (HSPIII) of the German Academic Exchange Service.

I am grateful to my supervisors for their help in making my three years' stay at the Technical University of Denmark at all possible. In particular, many thanks to Tom Høholdt for always being at my disposal when questions nagged, for his encouragement and optimism, and for giving me the opportunity to travel the world and meet coding theorists.

Other people have influenced the time of my studies. I would like to thank Professor Peter Cameron for his hospitality at Queen Mary, University of London two weeks in the spring of 2000 and his help in clearing the path for my future there, Jørgen Bo Christensen for helping me back on my feet half way through, my fellow Ph.D. students for companionship, in particular Thomas Jakobsen for a few intense months working inspiringly together, Wolfhard Kliem for Die Zeit and Robert Sinclair for draft reading and being around.

Last but not least, thanks to Carl Edward Rasmussen for believing in me and enduring me.

Agnes Eileen Heydtmann,
31st of January, 2001, Lyngby

CONTENTS

1. INTRODUCTION

The theory of error-correcting codes is a relatively young science that came into life as information technology evolved. This is what the following focuses on although in one chapter the motivation for the work came from cryptology, the science of reading and writing secret code.

Error-correcting codes are used in communication channels such as satellite communication as well as data storage systems like compact discs. The classical approximate model for both of these setups is the binary symmetric channel. Data represented as a sequence of bits, is thought of as being transmitted from some source via this channel to a receiver. Due to noise, individual bits are altered with probability p and are received correctly with probability $1 - p$. Suppose the sequence of bits to be sent is divided into strings of k bits. There are 2^k distinct strings which are assumed to appear with equal probability in the sequence. Error-correcting codes are used as a scheme to add redundancy to these strings, i.e. to add a number of bits to each length-k string in order to make the strings more distinguishable. This will be helpful for reconstructing when bits are altered during transmission via the channel.

Thus, k so called information bits are encoded into $n \geq k$ bits. The set of the 2^k bit strings of length n form the code, also referred to as the block code, with its elements called codewords. When codewords are transmitted over a binary symmetric channel each bit in the word is altered with probability p. Therefore, at the receiving end there must be a decoder that converts each of the possibly faulty words back into the most likely codeword it might have originated in. "Most likely" in this context means a codeword closest to the received word given some distance measure in the space of binary n-tuples. The most natural distance measure is the Hamming distance which computes the distance between two words as the number of differing corresponding entries. This definition also gives rise to the minimum distance d of a code, i.e. the smallest distance between two codewords. If less than $d/2$ errors occurred under transmission, then there is only one codeword closest to the received word. In that case, this is the word that was most likely sent. Once the received word has been decoded to a codeword, it can be translated back into a string of k information bits. However if no redundancy is added, that is if $n = k$, then $d = 1$ and there is no space for a better estimate than the received string. Faulty words cannot be distinguished from correct ones.

Both numbers, k and d should be judged in relation to the block length n of the code. Therefore two more quantities are introduced: the rate $R = k/n$, serving as a measure of the efficiency of the code, and the relative minimum distance of the code $\delta = d/n$, measuring the error-correcting capacity of the code. When comparing different types of codes of the same rate and thereby of

equal efficiency, another quality measure is the probability of incorrect decoding in a maximum likelihood decoding algorithm. Maximum likelihood decoding refers to the principle that upon receiving an arbitrary binary string of length n, the algorithm returns a codeword that was most likely sent, i.e. one that is closest. This probability depends on the bit error probability p. The smaller the probability of incorrect decoding given p the better the code. In concrete cases the probability of incorrect decoding is difficult to calculate, but when p is small, a good approximation and in any case an upper bound is the number

$$1 - \sum_{i=1}^{\lfloor \frac{d-1}{2} \rfloor} \binom{n}{i} p^i (1-p)^{n-i}.$$

This makes intuitive sense, since given a received word altered in less than $d/2$ positions, the probability of incorrect decoding is 0.

Given the probability p, another number, called the capacity, is associated with the binary symmetric channel. It plays a central role in Shannon's theorem published in his famous paper *A Mathematical Theory of Communication* (Shannon 1948) that marks the beginning of coding theory. The theorem says that for any $\epsilon > 0$, no matter how small, any R less than the capacity of the channel and n sufficiently large, there exists a code of length n and rate $\geq R$ such that the probability of incorrect decoding is $< \epsilon$.

This result tells us something about the asymptotic behavior of codes, that is about their behavior considering a sequence of codes with increasing block length. It is surprising and in some sense counterintuitive. The simplest code that can be imagined, a repetition code, does not live up to the theorem. In such a code the information bits are simply repeated r times to make a code of length $n = rk$. As r increases, so does the length n and the probability of incorrect decoding converges to 0. But also the rate $R = 1/r$ converges to 0. The theorem claims that there exist sequences of codes of increasing length such that the probability of incorrect decoding converges to 0 while their rate is bounded away from 0. In fact it claims that there exist sequences of such codes with rate arbitrarily close to the capacity of the channel — these codes are said to reach capacity. The price to pay for such good codes is long code lengths. The longer the code the bigger the delay at the receiving end.

The proof of Shannon's theorem is not constructive, but uses probabilistic methods, and a large research territory was discovered at the same time. A lot of effort has been directed towards finding good codes like the theorem promises. But low probability of incorrect decoding is not the only property to strive for. For practical reasons codes should among many things also have some structure to facilitate encoding and decoding.

Algebraic block codes are such codes with some regularity. They are based on algebraic structures and their encoding and decoding algorithms rely on the knowledge of or require new insight into those algebraic structures. But most of the different types of algebraic block codes known do not live up to Shannon's theorem. When considering sequences of one type of algebraic codes with increasing block length, either the rate converges to 0 or the relative minimum distance does, which indicates that the probability of incorrect decoding in-

creases. Justesen codes were the first algebraic codes to be discovered where sequences of codes with increasing length exist such that both R and δ could be bound away from 0. This property implies that the codes are good in the sense of Shannon's theorem - the probability of incorrect decoding can be chosen arbitarily small at non-zero rate up to some maximum rate, possibly not quite the capacity of the channel.

However, since relatively short block length, a simple decoding algorithm and high efficiency of the code are similarly desirable, codes that do not reach capacity can be good enough for practical applications. This is one of the reasons why codes like Reed-Solomon codes, Reed-Muller codes or quadratic residue codes are still heavily studied. Also, equally important, there is the hope that a fuller understanding of these codes will give rise to the construction of related, but better codes in the sense of Shannon's theorem and other desired properties.

The ideas described above can be generalized to codes over any finite field instead of only bits. The present work focuses on decoding algorithms for different types of algebraic error-correcting codes over an arbitrary finite field. All of the following chapters are a compilation of four papers that my research work of the past three years resulted in.

In Chapter 2, two well known decoding algorithms for decoding alternant codes are compared, the famous Berkekamp-Massey algorithm (Berlekamp 1968; Massey 1969) and the Euclidean algorithm (Sugiyama et al. 1975). To some few members of the coding theory community some of the similarities of the two algorithms were known (Dornstetter 1987). However, here is presented what can be considered a full record of the similarities and differences of the two algorithms. The algorithms are equivalent in the sense that all elementary calculations performed are the same, the intermediate results correspond directly to each other and the sequence of steps in one can be traced as a sequence of steps in the other. Special effort has been directed towards readability and the intuition of the reader by illustrations through the Fundamental Iterative Algorithm (Feng and Tzeng 1991). A shortened version of the chapter treating the case of Reed-Solomon codes has been published as correspondence in the November, 2000 issue of the *IEEE Transactions on Information Theory*.

We have seen above that the minimum distance d of a code can be regarded as an alternative quality measure for a code. High minimum distance compared to the block length is desirable as words are more distinguishable the further they are apart and it minimizes the probability of incorrect decoding. Until recently, decoding algorithms for algebraic codes were at most able to decode less than $d/2$ errors. That is, if $d/2$ or more errors occurred, then the algorithm could at most detect errors, maybe even decode incorrectly. The reason is of course that if that many errors occur two or more codewords may lie closest to the received word. A few years ago, Sudan (1997) developed an algorithm for Reed-Solomon codes that decodes beyond half the minimum distance by returning a list of closest codewords. This is interesting, particularly if at some stage between decoder and receiver it could be determined — out of context, for example — which of the list elements was most likely sent. This does not make d a less relevant parameter, since even decoding beyond half the minimum

distance still leaves scope for mistakes. It was this work by Sudan that gave inspiration for Chapter 3. Here, his algorithm is generalized to Reed-Muller codes. To this extent a boolean polynomial is interpolated and factored despite the fact that the ring of boolean polynomials is not a unique factorization domain. Under the assumption that the weights of a given Reed-Muller code are approximately binomially distributed, the error-correcting capability of the algorithm is analyzed. The paper will be published in the Proceedings of the Fifth International Conference on Finite Fields and Applications that took place in Augsburg, Germany, 1999 to appear in March 2001.

Linear block codes, i.e. those codes that can be regarded as a vector space over a finite field, are usually compared to the Gilbert-Varshamov bound. This is a lower bound on the maximum rate a code of a given relative minimum distance can reach asymptotically. So, a code performing well with respect to this bound does well in the sense of Shannon's theorem. Many algebraic codes meet this bound, and for a long time it was conjectured that it was not just a lower bound, but also an upper one. However in 1982 Tsfasman et al. proved the existence of geometric Goppa codes that were better than the Gilbert-Varshamov bound. This spurred a sudden interest in these codes and finally Garcia and Stichtenoth (1995) were able to find concrete sequences of such geometric Goppa codes defined over sufficiently large finite fields. The last two chapters can be regarded as an extension of this interest in geometric Goppa codes to their generalization. The construction by Xing et al. (1999) which uses this generalization in a concatenated fashion, revealed some promising examples of new codes with improved parameters (Ding et al. 2000).

Generalized geometric Goppa codes are formed by evaluating functions of an algebraic function field in places of arbitrary degree. This results in a tuple with entries in different extension fields of the base field. Chapter 4 investigates the structure of these codes. The codes are related to conventional geometric Goppa codes and parallels are drawn. The dual of a generalized geometric Goppa code is defined, and it is shown that it can be represented in terms of Weil differentials and that it belongs to the same family of codes. Finally, a decoding algorithm is presented that can also be used to decode the mentioned concatenated codes. It is the basic algorithm for decoding conventional geometric Goppa codes that has been adapted to the new setting. An paper with the same content has been submitted to the journal *Communications in Algebra*.

In Chapter 5, Sudan's improved algorithm for decoding geometric Goppa codes beyond half the minimum distance (Guruswami and Sudan 1999) is applied to generalized geometric Goppa codes. It comes as no surprise that under certain circumstances this algorithm can correct more errors than the decoding algorithm suggested in the previous chapter. Certainly, this algorithm can also be applied when decoding the concatenated codes by Xing et al. (1999). Additionally, an algorithm to obtain so called increasing zero bases of spaces of functions of an algebraic function field is developed. Such bases are necessary for the described decoding algorithm. This paper has been submitted to the journal *Finite Fields and Their Applications*.

Note that each of the following chapters have their own list of references and that all references are gathered in the bibliography at the end of the thesis.

2. ON THE EQUIVALENCE OF THE BERLEKAMP-MASSEY AND THE EUCLIDEAN ALGORITHM FOR DECODING

Agnes E. Heydtmann* and Jørn M. Jensen

Department of Mathematics, Building 303, Technical University of Denmark,
DK-2800 Kongens Lyngby, Denmark. {Agnes.Heydtmann,J.M.Jensen}@mat.dtu.dk

Abstract

The Berlekamp-Massey algorithm and the Euclidean algorithm for decoding have been considered as two different algorithms for solving the same problem, namely the one given by the key equation. In this paper we argue that they are essentially identical by showing how one can be adapted to perform the same arithmetics as the other. As a tool we use Feng and Tzeng's Fundamental Iterative Algorithm which when adapted to the syndrome matrix is regarded as equivalent to the Berlekamp-Massey algorithm.

2.1 Introduction

Decoding alternant codes such as BCH, Reed-Solomon and Goppa codes consists essentially of solving the so called key equation. Both the Berlekamp-Massey algorithm (BMA) (Berlekamp 1968; Massey 1969) and the Euclidean algorithm for decoding (EA) by Sugiyama et al. (1975) serve this purpose in seemingly different ways, but whereas the first is more efficient (compare Sections 2.3.5 and 2.4.3), the latter has been considered simpler. Dornstetter (1987) uncovered some of the parallels of the two algorithms, but still many think of these by now adopted as classical decoding algorithms as two different methods for solving one problem. Later Feng and Tzeng (1991) developed the Fundamental Iterative Algorithm (FIA). This algorithm can manipulate matrices in a general setting, but when used on the syndrome matrix — and we will use the name FIA in this sense — its equivalence to the BMA is widely accepted. However the FIA is of great tutorial value as it basically performs a kind of Gaussian elimination on the syndrome matrix which makes it accessible to the intuition of the reader. In our opinion the BMA is much easier to comprehend and to work with in the version of the FIA, and we will make use of that when

*During this work Agnes E. Heydtmann was supported by a graduate student scholarship (HSPIII) of the German Academic Exchange Service.

demonstrating the equivalence. Through the usage of the FIA it will become apparent that every single coefficient in the treated syndrome matrix and the different versions of the locator polynomial in the BMA correspond to coefficients of the various polynomials of the repeated divisions performed by the EA, and for every step in one algorithm there is a corresponding one in the other.

In order to show which parts of the two algorithms correspond precisely to each other, the FIA is extended to compute the error evaluator and coevaluator simultaneously to the error locator. This increases its complexity of course, but it makes the FIA equivalent to and as complex as the EA.

Further, it is shown how in the EA some steps can be eliminated such that it either calculates the error evaluator or the error coevaluator. This way the EA becomes equivalent to and as complex as the FIA extended by either error evaluator or coevaluator and thus equivalent and as complex as the original BMA. Dornstetter (1987) also eliminated steps from the EA such that it became equivalent to the BMA. However it is not described in that paper that the removed steps correspond to the removal of the calculation of the error coevaluator (which is where his and the present work coincide), but instead it is claimed that both algorithms compute the same thing, whereas one is more complex than the other.

This paper has been written in the hope of being fairly self-contained focusing on the intuition of every step. Section 2.2 introduces notation and the setting. Section 2.3 presents the BMA in the version of the FIA and the extension to the calculation of the error evaluator and coevaluator in addition to the error locator. In Section 2.4 the EA is adapted in two steps to increasingly resemble the BMA and we discuss which parts of the two algorithms correspond to each other. Section 2.5 is reserved for some remarks on the changes made and conclusions will be drawn in Section 2.6.

Simulation of the discussed decoding processes has been done with the computer algebra system SINGULAR (Greuel et al. 1998).

2.2 Preliminaries

Let $\mathbf{a} = (a_1, \ldots, a_n)$ with a_i distinct elements of the field $\mathbb{F}_{q^m}$ and similarly $\mathbf{y} = (y_1, \ldots, y_n)$ with y_i non-zero elements of $\mathbb{F}_{q^m}$. The alternant code $\mathcal{A}_k(\mathbf{a}, \mathbf{y})$, $k \in \mathbb{N}$ is the subfield subcode over $\mathbb{F}_q$ of the generalized Reed-Solomon code $\mathcal{GRS}_k(\mathbf{a}, \mathbf{y})$ over $\mathbb{F}_{q^m}$ which has parity check matrix

$$H = \begin{pmatrix} y_1 & y_2 & \cdots & y_n \\ a_1 y_1 & a_2 y_2 & \cdots & a_n y_n \\ a_1^2 y_1 & a_2^2 y_2 & \cdots & a_n^2 y_n \\ \vdots & \cdots & & \cdots \\ a_1^{n-k-1} y_1 & a_2^{n-k-1} y_2 & \cdots & a_n^{n-k-1} y_n \end{pmatrix}.$$

In other words

$$\mathcal{A}_k(\mathbf{a}, \mathbf{y}) = \{\mathbf{c} \in \mathbb{F}_q^n : H\mathbf{c}^t = 0\}.$$

It is an $[n, k', d]$ code with $n - m(n-k) \leq k' \leq k$ and $d \geq n - k + 1$. For alternant codes in general compare for example with (MacWilliams and Sloane 1977, Chap. 12).

Suppose the codeword $\mathbf{c} = (c_1, c_2, \ldots, c_n) \in \mathcal{A}_k(\mathbf{a}, \mathbf{y})$ is sent but the word $\mathbf{r} = \mathbf{c} + \mathbf{e} = (r_1, r_2, \ldots, r_n)$ is received with error $\mathbf{e} = (e_1, e_2, \ldots, e_n)$. The goal is to decode $\mathbf{r}$ to $\mathbf{c}$ which involves finding $\mathbf{e}$. We assume that the number of errors ϵ is bounded by $(n-k)/2$ and suppose errors occurred in locations $l_1, \ldots, l_\epsilon$. Usually the syndrome of the received word $\mathbf{r}$ is the vector $H\mathbf{r}^t$, but in the context of decoding alternant codes the *syndromes* of $\mathbf{r}$ are the components of the usual syndrome, ie.

$$S_i = \sum_{j=1}^{n} a_j^i y_j r_j = \sum_{j=1}^{n} a_j^i y_j (c_j + e_j) = \sum_{j=1}^{n} a_j^i y_j e_j = \sum_{j=1}^{\epsilon} a_{l_j}^i y_{l_j} e_{l_j}, \ 0 \leq i \leq n-k-1.$$

The *syndrome polynomial* is

$$S = S_0 + S_1 x + \cdots + S_{n-k-1} x^{n-k-1}.$$

We define further the *error locator polynomial*

$$\Lambda = \prod_{i=1}^{\epsilon} (1 - a_{l_j} x) = \Lambda_\epsilon x^\epsilon + \cdots + \Lambda_1 x + \Lambda_0,$$

the *error evaluator polynomial*

$$\Omega = \sum_{j=1}^{\epsilon} y_{l_j} e_{l_j} \prod_{\substack{i=1 \\ i \neq j}}^{\epsilon} (1 - a_{l_i} x) = \Omega_{\epsilon-1} x^{\epsilon-1} + \cdots + \Omega_1 x + \Omega_0$$

and what Pretzel (1992, pp. 244, 274) calls the *error coevaluator polynomial* in the context of BCH codes

$$\Psi = \sum_{j=1}^{\epsilon} a_{l_j}^{n-k} y_{l_j} e_{l_j} \prod_{\substack{i=1 \\ i \neq j}}^{\epsilon} (1 - a_{l_i} x) = \Psi_{\epsilon-1} x^{\epsilon-1} + \cdots + \Psi_1 x + \Psi_0.$$

It is obvious that Λ and Ω as well as Λ and Ψ do not have any common factors. Also $\Lambda(0) = \Lambda_0 = 1$ and $\deg \Lambda \leq \lfloor (n-k)/2 \rfloor$ and $\deg \Omega, \deg \Psi < \lfloor (n-k)/2 \rfloor$. For the remainder of this chapter let $t = \lfloor (n-k)/2 \rfloor$ and $t' = \lceil (n-k)/2 \rceil$. The following theorem shows how the four polynomials S, Λ, Ω and Ψ are related.

Theorem 2.1 *Let S be the syndrome polynomial of a received word corresponding to no more than t errors. Then the error locator polynomial Λ, the error evaluator polynomial Ω and the error coevaluator polynomial Ψ are the unique solutions to the equation*

$$\Lambda \cdot S = \Omega - \Psi x^{n-k} \tag{2.1}$$

with the properties $\gcd(\Lambda, \Omega) = 1$, $\gcd(\Lambda, \Psi) = 1$, $\Lambda(0) = 1$, $\deg \Lambda \leq t$ *and* $\deg \Omega, \deg \Psi < t$. *In particular if λ, ω, ψ is another corresponding solution with,* $\deg \lambda \leq t$ *and* $\deg \omega, \deg \psi < t$, *then $\lambda = \phi\Lambda$, $\omega = \phi\Omega$, $\psi = \phi\Psi$ for some polynomial ϕ.*

Proof : The polynomials Λ, Ω and Ψ are solutions to (2.1) by the following calculation:

$$\begin{aligned}
\frac{\Omega - \Psi x^{n-k}}{\Lambda} &= \frac{\sum_{j=1}^{\epsilon} y_{l_j} e_{l_j} (1 - a_{l_j}^{n-k} x^{n-k}) \prod_{\substack{i=1 \\ i \neq j}}^{\epsilon} (1 - a_{l_i} x)}{\prod_{i=1}^{\epsilon} (1 - a_{l_j} x)} \\
&= \sum_{j=1}^{\epsilon} \frac{y_{l_j} e_{l_j} (1 - a_{l_j}^{n-k} x^{n-k})}{1 - a_{l_j} x} \\
&= \sum_{j=1}^{\epsilon} y_{l_j} e_{l_j} (1 - a_{l_j}^{n-k} x^{n-k}) \sum_{i=0}^{\infty} a_{l_j}^{i} x^{i} \\
&= \sum_{i=0}^{\infty} \sum_{j=1}^{\epsilon} a_{l_j}^{i} y_{l_j} e_{l_j} x^{i} (1 - a_{l_j}^{n-k} x^{n-k}) \\
&= \sum_{i=0}^{n-k-1} \sum_{j=1}^{\epsilon} a_{l_j}^{i} y_{l_j} e_{l_j} x^{i} + \sum_{i=n-k}^{\infty} \sum_{j=1}^{\epsilon} a_{l_j}^{i} y_{l_j} e_{l_j} x^{i} \\
&\qquad - \sum_{i=0}^{\infty} \sum_{j=1}^{\epsilon} a_{l_j}^{n-k+i} y_{l_j} e_{l_j} x^{n-k+i} \\
&= S
\end{aligned}$$

Suppose now that λ, ω and ψ be another solution to (2.1) with $\deg \lambda \leq t$ and $\deg \omega, \deg \psi < t$. Thus

$$\Lambda \cdot S = \Omega - \Psi x^{n-k} \qquad \text{and} \qquad \lambda \cdot S = \omega - \psi x^{n-k}$$

and by multiplication with λ and Λ respectively we obtain

$$\lambda\Omega - \lambda\Psi x^{n-k} = \Lambda\omega - \Lambda\psi x^{n-k}.$$

But both $\lambda\Omega$ and $\Lambda\omega$ are polynomials of degree $< n-k$ by the degrees of the polynomials involved and $\lambda\Psi x^{n-k}$ and $\Lambda\psi x^{n-k}$ ones of degree $\geq n-k$. Therefore $\lambda\Omega = \Lambda\omega$. Since $\gcd(\Lambda, \Omega) = 1$ it follows that $\lambda = \phi\Lambda$ and $\omega = \phi\Omega$ for some polynomial ϕ. But as both sets of polynomials fulfill equation (2.1), we also obtain $\psi = \phi\Psi$. The uniqueness property follows. □

We call equation (2.1) the *key equation* although more commonly this refers to the related equation $\Lambda \cdot S \equiv \Omega \bmod x^{n-k}$.

Theorem 2.1 shows that if there is an algorithm that can find Λ, Ω and Ψ with the properties $\gcd(\Lambda, \Omega) = 1$, $\gcd(\Lambda, \Psi) = 1$, $\Lambda(0) = 1$, $\deg \Lambda \leq t$ and $\deg \Omega, \deg \Psi < t$ and fulfilling the key equation (2.1), then they must be the error locator, evaluator and coevaluator respectively.

Finding Λ will result in obtaining the error locations by either factorization or a brute force search for zeroes in the finite field $\mathbb{F}_{q^m}$. Once Λ is found Ω and/or Ψ can easily be computed by the key equation (2.1). Let

$$\Lambda' = -\sum_{j=1}^{\epsilon} a_{l_j} \prod_{\substack{i=1 \\ i \neq j}}^{\epsilon} (1 - a_{l_i} x),$$

which is the formal derivative of Λ. With the help of one of Ω or Ψ, error values can then be obtained with the so called Forney algorithm

$$e_{l_i} = -\frac{a_{l_i}\Omega(a_{l_i}^{-1})}{y_{l_i}\Lambda'(a_{l_i}^{-1})} = -\frac{\Psi(a_{l_i}^{-1})}{a_{l_i}^{n-k-1}y_{l_i}\Lambda'(a_{l_i}^{-1})}. \tag{2.2}$$

In this way we get all components of $\mathbf{e}$.

Summarizing it can be said that in decoding which consists of finding $l_1, \ldots, l_\epsilon$ and $e_{l_1}, \ldots, e_{l_\epsilon}$ accordingly, three steps are involved:

1. Calculating the syndromes S_i, $0 \leq i \leq n-k$,
2. given the syndrome polynomial S, finding the error locator Λ and error evaluator Ω usually with the BMA, see for example Berlekamp (1968) and Massey (1969), or the EA by Sugiyama et al. (1975) which also calculates the error coevaluator Ψ,
3. given Λ and Ω or Ψ, computing the actual error locations by either factorization of Λ or a brute force search of its roots in $\mathbb{F}_{q^m}$, and then values by the Forney algorithm (2.2).

In the next sections we are focusing on step 2 which is the computationally most expensive part.

2.3 The Berlekamp-Massey Algorithm

2.3.1 A Matrix Approach to the Key Equation

We can rephrase (2.1) in matrix form as

$$\begin{pmatrix} 0 & \cdots & 0 & S_0 \\ \vdots & \iddots & \iddots & \vdots \\ 0 & S_0 & \cdots & S_{\epsilon-1} \\ S_0 & S_1 & \cdots & S_\epsilon \\ S_1 & S_2 & \cdots & S_{\epsilon+1} \\ \vdots & \vdots & & \vdots \\ S_{n-k-\epsilon-1} & S_{n-k-\epsilon} & \cdots & S_{n-k-1} \\ \vdots & & \iddots & 0 \\ & & \iddots & \vdots \\ S_{n-k-1} & 0 & \cdots & 0 \end{pmatrix} \begin{pmatrix} \Lambda_\epsilon \\ \Lambda_{\epsilon-1} \\ \vdots \\ \Lambda_1 \\ 1 \end{pmatrix} = \begin{pmatrix} \Omega_0 \\ \vdots \\ \Omega_{\epsilon-1} \\ 0 \\ 0 \\ \vdots \\ 0 \\ -\Psi_0 \\ \vdots \\ -\Psi_{\epsilon-1} \end{pmatrix}. \tag{2.3}$$

The terms corresponding to $x^\epsilon, \ldots, x^{n-k-1}$ in the product $\Lambda \cdot S$ are 0 and as we will see this information is already enough to find Λ. Consider the middle part of (2.3):

$$\begin{pmatrix} S_0 & S_1 & \cdots & S_\epsilon \\ S_1 & S_2 & \cdots & S_{\epsilon+1} \\ \vdots & \vdots & & \vdots \\ S_{n-k-\epsilon-1} & S_{n-k-\epsilon} & \cdots & S_{n-k-1} \end{pmatrix} \begin{pmatrix} \Lambda_\epsilon \\ \Lambda_{\epsilon-1} \\ \vdots \\ \Lambda_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{2.3'}$$

The coefficients of Λ give a linear dependence of the column number $\epsilon+1$ in (2.3') in terms of the previous columns. However ϵ is also an unknown and instead we study the $t' \times (t+1)$ *syndrome matrix*

$$\mathbf{S} = \begin{pmatrix} S_0 & S_1 & \cdots & S_\epsilon & \cdots & S_t \\ S_1 & S_2 & \cdots & S_{\epsilon+1} & \cdots & S_{t+1} \\ \vdots & \vdots & & \vdots & & \vdots \\ S_{t'-1} & S_{t'} & \cdots & S_{t'+\epsilon-1} & \cdots & S_{n-k-1} \end{pmatrix}. \tag{2.4}$$

Comparing with the matrix in (2.3') shows that the first $\epsilon+1$ columns are linearly dependent in $\mathbf{S}$ and the two matrices are identical if $\epsilon = t$.

By $\lambda = \lambda_{\nu-1}x^{\nu-1} + \cdots + \lambda_1 x + \lambda_0$ describing a linear combination of the first ν columns $\mathbf{A}_1, \ldots, \mathbf{A}_\nu$ of a matrix $\mathbf{A}$, we mean the combination $\lambda_{\nu-1}\mathbf{A}_1 + \cdots + \lambda_1\mathbf{A}_{\nu-1} + \lambda_0\mathbf{A}_\nu$. By such λ describing a linear dependence of the first ν columns of a matrix $\mathbf{A}$, we naturally mean that the above linear combination is 0. Proposition 2.1 describes that by finding such a linear dependence of the first $\epsilon+1$ columns of $\mathbf{S}$ the error locator Λ is found.

Proposition 2.1 *Any $\lambda = \lambda_\epsilon x^\epsilon + \cdots + \lambda_1 x + \lambda_0$ describing a linear dependence of the leading $\epsilon+1$ columns $\mathbf{S}_1, \ldots, \mathbf{S}_{\epsilon+1}$ of the syndrome matrix $\mathbf{S}$ equals the error locator polynomial Λ up to a constant factor. In particular $\lambda_\epsilon \neq 0$.*

The proof is very similar to the one for the uniquness statement in Theorem 2.1.

Proof : Let λ be as above. Since the linear dependency implies that the coefficients corresponding to $x^\epsilon, \ldots, x^{t'+\epsilon-1}$ in the product $\lambda \cdot S$ are 0, there exist some polynomials ψ, ω with $\deg\omega < \epsilon$ such that

$$\lambda \cdot S = \omega - \psi x^{t'+\epsilon}.$$

The key equation (2.1) implies

$$\Lambda \cdot S = \Omega - \Psi x^{t-\epsilon} x^{t'+\epsilon}$$

and multiplying each equation with Λ and λ respectively gives

$$\Lambda\omega - \Lambda\psi x^{t'+\epsilon} = \lambda\Omega - \lambda\Psi x^{t-\epsilon}x^{t'+\epsilon}.$$

Since $\Lambda\omega$ and $\lambda\Omega$ are both of degree less than 2ϵ and the rest of the equation consists of terms $\geq t'+\epsilon$ it can be conclued that $\Lambda\omega = \lambda\Omega$. Thus $\Lambda \mid \lambda\Omega$. But $\gcd(\Lambda, \Omega) = 1$, therefore $\Lambda \mid \lambda$ and $\deg\lambda = \epsilon$. The claim follows. □

Thus, any algorithm that produces a linear dependence between as few initial columns of the syndrome matrix $\mathbf{S}$ as possible can easily be extended to finding the error locator polynomial Λ. Once Λ is found Ω or Ψ can be obtained easily and thus (2.3) as well as the key equation (2.1) can be solved by such an approach.

The next section presents the algorithm by Feng and Tzeng (1991) that finds a linear dependence — if there is one — of as few initial columns as possible in an arbitrary matrix. This algorithm will then be adapted to the special case of the syndrome matrix.

2.3.2 The Fundamental Iterative Algorithm

Let $\mathbf{A}$ be an arbitrary matrix with columns $\mathbf{A}_i$. In order to find the smallest number of linearly dependent initial columns of $\mathbf{A}$ one can perform a kind of Gaussian elimination on $\mathbf{A}$: Start with the first column. By subtracting multiples of previous columns from the current column obtain columns with as many initial zeros as possible until one column consists entirely of zeros. An example of the result of such a process could be the matrix

$$\begin{pmatrix} \delta_1 & 0 & 0 & 0 & 0 & 0 & 0 & \\ & \delta_2 & 0 & 0 & 0 & 0 & 0 & \\ & & 0 & 0 & \delta_5 & 0 & 0 & \\ & & 0 & \delta_4 & & 0 & 0 & \mathbf{A}_8 \\ & & \delta_3 & & & 0 & 0 & \\ & & & & & \delta_6 & 0 & \\ & & & & & & 0 & \end{pmatrix}$$

where the δ_j's stand for non-zero entries and the empty components for arbitrary ones. Suppose in column number j all non-zero entries could be eliminated. By the process involved it is clear that the original columns $\mathbf{A}_1, \ldots, \mathbf{A}_{j-1}$ are linearly independent, that columns $\mathbf{A}_1, \ldots, \mathbf{A}_j$ are linearly dependent and the linear dependence is found by keeping track of the operations.

This describes the FIA developed by Feng and Tzeng (1991) and a formal outline is given in Algorithm 2.1. Note that the linear combinations and thereby also the final linear dependency are given in terms of a polynomial. This is motivated by the fact that we want to use the FIA on the syndrome matrix when finding the error locator polynomial.

To understand the formula (2.5) updating $\lambda^{(\nu)}$, suppose the FIA reached column number ν when performed on matrix $\mathbf{A}$ with entries $a_{\mu\nu}$. At that point each

$$\lambda^{(j)} = \lambda_{j-1}^{(j)} x^{j-1} + \cdots + \lambda_1^{(j)} x + \lambda_0^{(j)}, \;\; 1 \leq j \leq \nu$$

describes the linear combination

$$\tilde{\mathbf{A}}_j = \lambda_{j-1}^{(j)} \mathbf{A}_1 + \cdots + \lambda_1^{(j)} \mathbf{A}_{j-1} + \lambda_0^{(j)} \mathbf{A}_j \neq 0, \;\; 1 \leq j \leq \nu. \tag{2.6}$$

Note that if no initial entries in the ν-th column have been eliminated yet then $\lambda^{(\nu)} = 1$ and that in general $\lambda^{(j)}(0) = 1$, $1 \leq j \leq \nu$. By subtracting multiples of the $\tilde{\mathbf{A}}_j$ as many initial entries in column $\tilde{\mathbf{A}}_\nu$ as possible should be eliminated. Suppose $\tilde{\mathbf{A}}_\nu$ has initial zeros up to the μ_ν-th entry

$$\delta = \sum_{j=0}^{\nu-1} \lambda_j^{(\nu)} a_{\mu_\nu \, \nu-j} \neq 0$$

which we call a *discrepancy* from zero in row μ_ν. In order to eliminate δ a multiple of $\tilde{\mathbf{A}}_j$ which has the same number of initial zeroes followed by a discrepancy $\delta_j \neq 0$ in row $\mu_j = \mu_\nu$, should be subtracted. The new $\tilde{\mathbf{A}}_\nu$ is

Input: matrix $\mathbf{A} = (a_{\mu\nu})_{1\le\mu\le\mu_{\max},1\le\nu\le\nu_{\max}}$
Output: if possible $\lambda = \lambda_d x^d + \cdots + \lambda_1 x + \lambda_0$ with $\lambda_0 = 1$ describing a linear dependence between the lowest number $d+1$ of linearly dependent initial columns of $\mathbf{A}$

1: Start with row index $\mu_1 = 1$ and column index $\nu = 1$.
2: Initialize $\lambda^{(1)} = 1$.
3: **repeat**
4: Calculate discrepancy
$$\delta = \sum_{j=0}^{\nu-1} \lambda_j^{(\nu)} a_{\mu_\nu\,\nu-j}.$$
5: **if** $\delta \neq 0$ **then**
6: **if** $\mu_\nu = \mu_j$ for some $1 \le j < \nu$ **then**
7: Update $\lambda^{(\nu)}$ to
$$\lambda^{(\nu)} - \frac{\delta}{\delta_j}\lambda^{(j)} x^{\nu-j}. \tag{2.5}$$
8: Increment μ_ν by 1.
9: **else** $\{\mu_\nu \neq \mu_j$ for all $1 \le j < \nu\}$
10: Set $\delta_\nu = \delta$.
11: Increment ν by 1 and set $\mu_\nu = 1$.
12: Initialize $\lambda^{(\nu)} = 1$.
13: **end if**
14: **else** $\{\delta = 0\}$
15: Increment μ_ν by 1.
16: **end if**
17: **until** $\mu_\nu > \mu_{\max}$ or $\nu > \nu_{\max}$
18: **if** $\mu_\nu > \mu_{\max}$ **then**
19: **return** $\lambda^{(\nu)}$
20: **end if**

Algorithm 2.1: The Fundamental Iterative Algorithm

thereby going to be

$$\begin{aligned}
\tilde{\mathbf{A}}_\nu - \frac{\delta}{\delta_j}\tilde{\mathbf{A}}_j &\overset{(2.6)}{=} \lambda_{\nu-1}^{(\nu)}\mathbf{A}_1 + \cdots + \lambda_1^{(\nu)}\mathbf{A}_{\nu-1} + \lambda_0^{(\nu)}\mathbf{A}_\nu \\
&\quad - \frac{\delta}{\delta_j}\left(\lambda_{j-1}^{(j)}\mathbf{A}_1 + \cdots + \lambda_1^{(j)}\mathbf{A}_{j-1} + \lambda_0^{(j)}\mathbf{A}_j\right) \\
&= \left(\lambda_{\nu-1}^{(\nu)} - \frac{\delta}{\delta_j}\lambda_{j-1}^{(j)}\right)\mathbf{A}_1 + \cdots + \left(\lambda_{\nu-j}^{(\nu)} - \frac{\delta}{\delta_j}\lambda_0^{(j)}\right)\mathbf{A}_j \\
&\quad + \lambda_{\nu-j-1}^{(\nu)}\mathbf{A}_{j+1} + \cdots + \lambda_1^{(\nu)}\mathbf{A}_{\nu-1} + \lambda_0^{(\nu)}\mathbf{A}_\nu
\end{aligned}$$

and the polynomial $\lambda^{(\nu)}$ should be updated as follows

$$\begin{aligned}
&\left(\lambda^{(\nu)}_{\nu-1} - \frac{\delta}{\delta_j}\lambda^{(j)}_{j-1}\right)x^{\nu-1} + \cdots + \left(\lambda^{(\nu)}_{\nu-j} - \frac{\delta}{\delta_j}\lambda^{(j)}_{0}\right)x^{\nu-j} \\
&\qquad + \lambda^{(\nu)}_{\nu-j-1}x^{\nu-j-1} + \cdots + \lambda^{(\nu)}_{1}x + \lambda^{(\nu)}_{0} \\
&\quad = \lambda^{(\nu)}_{\nu-1}x^{\nu-1} + \cdots + \lambda^{(\nu)}_{1}x + \lambda^{(\nu)}_{0} \\
&\qquad - \frac{\delta}{\delta_j}\left(\lambda^{(j)}_{j-1}x^{j-1} + \cdots + \lambda^{(j)}_{1}x + \lambda^{(j)}_{0}\right)x^{\nu-j} \\
&\quad = \lambda^{(\nu)} - \frac{\delta}{\delta_j}\lambda^{(j)}x^{\nu-j}
\end{aligned}$$

which is exactly what is done in Algorithm 2.1. The factor $x^{\nu-j}$ can be thought as "aligning" the two polynomials $\lambda^{(\nu)}$ and $\lambda^{(j)}$ such that the latter can "mend the defect" of the first. The above should have made apparent why it is not actually necessary to find the linear combinations of the column vectors $\mathbf{A}_j$. Calculating discrepancies and λ-polynomials as in Algorithm 2.1 is enough and once all non-zero entries could be eliminated in one column, the algorithm ends with a polynomial that describes the corresponding linear dependence.

2.3.3 Adjustment of the Fundamental Iterative Algorithm to the Syndrome Matrix

We want to make use of the FIA in finding the error locator polynomial. When this algorithm is applied to the syndrome matrix a linear dependence will be found in column $\epsilon + 1$ by Section 2.3.1. By the fact that the FIA operates in such a way that the constant term of the polynomial candidates are kept equal to 1 at all times and by Proposition 2.1 the error locator polynomial Λ will be the output. Recall however that the syndrome matrix has a special structure, a so called Hankel form, which the above algorithm does not take into account.

Let us investigate: For the remainder of this chapter let $\mathbf{S}$ denote a non-trivial syndrome matrix, i.e. there occurred $0 < \epsilon \leq t$ errors and by the key equation (2.1) and the degrees of the polynomials involved, the first column of $\mathbf{S}$ is not the zero-vector. Suppose the FIA is run on $\mathbf{S}$ and stalls in column ν, row μ with polynomial Λ. By "stalling" in row μ we mean that the Λ-polynomial can eliminate $\mu - 1$ initial entries in the respective column, but not the non-zero discrepancy δ in row μ — compare with Figure 2.1. We go to the next column $\nu + 1$. Here, it can be seen that the linear combination of columns $\nu + 1, \ldots, 2$ defined by Λ gives a vector where $\mu - 2$ initial entries could be eliminated, but in row $\mu - 1$ we get the same discrepancy δ, because of the structure of $\mathbf{S}$. So it is sensible to reuse Λ instead of restarting with the constant polynomial 1 as the original FIA does. Now either the algorithm stalled in row $\mu - 1$ before such that we can eliminate discrepancy δ and update or it did not, in which case we can move to the next column and use Λ again to eliminate $\mu - 3$ initial entries in column $\nu + 2$, but get discrepancy δ in row $\mu - 2$. In this way the algorithm finds a column where it can eliminate this discrepancy δ. If the FIA has previously stalled in column ν_{old}, row μ_{old} such that $\mu_{\text{old}} < \mu$ (i.e. it never

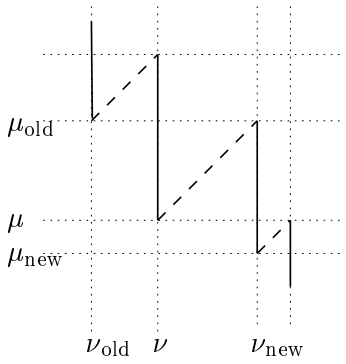


Figure 2.1: Illustration of the course of the adjusted FIA in column ν: Dotted lines represent certain rows and columns of the syndrome matrix respectively. The solid lines indicate where discrepancies are computed, the diagonal dashed lines mark jumps to new columns.

stalled in a row between μ_{old} and the current μ), we denote the corresponding versions of the error locator and discrepancy by Λ^{old} and δ_{old}. It turns out that a jump to column $\nu + \mu - \mu_{\text{old}}$ is going to be the right choice, because the discrepancy δ in row μ_{old} of this column can be eliminated by subtracting a multiple of $\tilde{\mathbf{S}}_{\nu_{\text{old}}}$ which denotes the linear combination of columns in $\mathbf{S}$ defined by Λ^{old}.

The following theorem formalizes these observations.

Theorem 2.2 *Let the FIA be run on the syndrome matrix* $\mathbf{S}$*. Suppose it stalls with polynomial* Λ *in column number* ν *with zero entries in* $\mu - 1 \geq \mu_{\text{old}}$ *initial positions and*

$$\delta = \sum_{i=0}^{\nu-1} \Lambda_i S_{\nu+\mu-i-2} \neq 0$$

as discrepancy in row μ *which cannot be eliminated, and it stalled previously in column* ν_{old}*, row* μ_{old} *with polynomial* Λ^{old} *and discrepancy* δ_{old} *such that* $\mu_{\text{old}} < \mu$ *and never stalled in a row between* μ_{old} *and* μ*. Then the* $\nu+\mu-\mu_{\text{old}}-1$ *initial columns of* $\mathbf{S}$ *are linearly independent and in the* $\nu_{\text{new}} = \nu + \mu - \mu_{\text{old}}$*-th column* $\mathbf{S}_{\nu_{\text{new}}}$ *non-zero entries in the* μ *initial rows can be eliminated in the following way:*

1. *the initial* $\mu_{\text{old}} - 1$ *entries by setting* $\tilde{\mathbf{S}}_{\nu_{\text{new}}}$ *to the linear combination corresponding to* $\Lambda^{\text{new}} = \Lambda$*,*
2. *the* μ_{old}*-th entry* $\delta \neq 0$ *by updating* $\tilde{\mathbf{S}}_{\nu_{\text{new}}}$ *and correspondingly* Λ^{new} *to*

$$\tilde{\mathbf{S}}_{\nu_{\text{new}}} - \frac{\delta}{\delta_{\text{old}}} \tilde{\mathbf{S}}_{\nu_{\text{old}}} \quad \text{and} \quad \Lambda^{\nu_{\text{new}}} - \frac{\delta}{\delta_{\text{old}}} \Lambda^{\text{old}} x^{\nu_{\text{new}} - \nu_{\text{old}}}, \tag{2.7}$$

3. *the* $\mu_{\text{old}} + 1$ *up to* μ*-th entry by finding non-zero discrepancies*

$$\delta_{\text{new}} = \sum_{i=0}^{\nu_{\text{new}}-1} \Lambda_i^{\text{new}} S_{\nu_{\text{new}}+\mu_{\text{new}}-i-2}, \quad \mu_{\text{old}} + 1 \leq \mu_{\text{new}} \leq \mu$$

and then eliminating them by updating $\tilde{\mathbf{S}}_{\nu_{\text{new}}}$ *and correspondingly* Λ^{new} *to*

$$\tilde{\mathbf{S}}_{\nu_{\text{new}}} - \frac{\delta_{\text{new}}}{\delta}\tilde{\mathbf{S}}_{\nu_{\text{new}}-(\mu_{\text{new}}-\mu_{\text{old}})} \quad \textit{and} \quad \Lambda^{\text{new}} - \frac{\delta_{\text{new}}}{\delta}\Lambda x^{\mu_{\text{new}}-\mu_{\text{old}}}. \tag{2.8}$$

Further, the above is also correct if in the beginning we set $\Lambda^{\text{old}} = 0$, $\delta_{\text{old}} = 1$, $\nu_{\text{old}} = 2$ *and* $\mu_{\text{old}} = 0$ *as well as* $\Lambda = 1$, $\nu = 1$ *and* $\mu = 1$, *and we obtain* $\nu_{\text{new}} = \nu + \mu - \mu_{\text{old}} = \mu + 1$ *in general.*

Proof: We start by proving the theorem for the initial conditions $\Lambda^{\text{old}} = 0$, $\delta_{\text{old}} = 1$, $\nu_{\text{old}} = 2$ and $\mu_{\text{old}} = 0$, $\Lambda = 1$, $\nu = 1$ and $\mu = 1$. The algorithm stalls with $\Lambda = 1$ in column 1, row μ such that $S_0 = \ldots = S_{\mu-1} = 0$ and $\delta = S_\mu \neq 0$. Because of the structure of $\mathbf{S}$ the first $\nu + \mu - \mu_{\text{old}} - 1 = \mu$ columns are linearly independent. In the $\nu + \mu - \mu_{\text{old}} = \mu + 1$-th column there are no updates to be made in rows $\mu_{\text{old}} - 1$ or μ_{old}, but the corresponding definitions of $\Lambda^{\text{new}} = \Lambda = 1$ are correct. Eliminating discrepancies in rows $\mu_{\text{old}} + 1 = 1$ up to μ should be done precisely as given in the theorem.

Now, as in the theorem, suppose that we stall in the ν-th column with polynomial Λ which eliminates $\mu - 1 \geq \mu_{\text{old}}$ initial components of $\mathbf{S}_\nu$ and $\delta = \sum_{i=0}^{\nu-1} \Lambda_i S_{\nu+\mu-i-2} \neq 0$ as μ-th entry which cannot be eliminated. Columns $\tilde{\mathbf{S}}_1, \ldots, \tilde{\mathbf{S}}_{\nu-1}$ in which as many initial entries as possible have been eliminated, are by the FIA linearly independent and therefore so are $\mathbf{S}_1, \ldots, \mathbf{S}_{\nu-1}$.

The fact that we are stalling in the ν-th column, μ-th row means that

$$\begin{aligned}
0 &= \Lambda_{\nu-1}S_0 + \cdots + \Lambda_1 S_{\nu-2} + S_{\nu-1} \\
0 &= \Lambda_{\nu-1}S_1 + \cdots + \Lambda_1 S_{\nu-1} + S_\nu \\
&\vdots \\
0 &= \Lambda_{\nu-1}S_{\mu-2} + \cdots + \Lambda_1 S_{\nu+\mu-4} + S_{\nu+\mu-3} \\
\delta &= \Lambda_{\nu-1}S_{\mu-1} + \cdots + \Lambda_1 S_{\nu+\mu-3} + S_{\nu+\mu-2}
\end{aligned}$$

in terms of the individual syndromes. Consequently the linear combinations

$$\Lambda_{\nu-1}\mathbf{S}_i + \cdots + \Lambda_1 \mathbf{S}_{\nu+i-2} + \mathbf{S}_{\nu+i-1}, \qquad 1 \leq i \leq \min\{t - \nu + 2, \mu\} \tag{2.9}$$

have zero entries in $\mu - i$ initial places and $\delta \neq 0$ in row $\mu - i + 1$, because of the structure of $\mathbf{S}$, and they are linearly independent. Note that there must exist an index i and linear combination as in (2.9) such that there are zero entries in $\mu_{\text{old}} - 1$ initial positions and $\delta \neq 0$ in row μ_{old} indicating a linear dependence on columns $\mathbf{S}_1, \ldots, \mathbf{S}_{\nu_{\text{old}}}$, since otherwise $\mathbf{S}$ would consist of linearly independent columns. This is impossible by our assumptions on $\mathbf{S}$. Therefore the choice $i = \mu - \mu_{\text{old}} + 1$ must be a valid one in (2.9).

Consider (2.9) for the cases $i = 1, \ldots, \mu - \mu_{\text{old}}$, then the corresponding combinations have zero entries in $\mu - 1, \ldots, \mu_{\text{old}}$ initial positions and then $\delta \neq 0$ in rows $\mu, \ldots, \mu_{\text{old}} + 1$ correspondingly. Set $\tilde{\mathbf{S}}_\nu, \ldots, \tilde{\mathbf{S}}_{\nu+\mu-\mu_{\text{old}}-1}$ to those linear combinations and by the fact that the linear combinations $\tilde{\mathbf{S}}_1, \ldots, \tilde{\mathbf{S}}_{\nu-1}$ corresponding to other versions of Λ have their first non-zero entry other places than

in rows $\mu_{\text{old}}+1,\ldots,\mu$ it is clear that columns $\tilde{\mathbf{S}}_1,\ldots,\tilde{\mathbf{S}}_{\nu+\mu-\mu_{\text{old}}-1}$ and therefore also $\mathbf{S}_1,\ldots,\mathbf{S}_{\nu+\mu-\mu_{\text{old}}-1}$ are linearly independent. This proves the first claim of the theorem.

Now let $i=\mu-\mu_{\text{old}}+1$ in (2.9). The linear combination has $\mu_{\text{old}}-1$ initial zeros and $\delta\neq 0$ as μ_{old}-th entry. Setting $\nu_{\text{new}}=\nu+\mu-\mu_{\text{old}}$ and $\tilde{\mathbf{S}}_{\nu_{\text{new}}}$ to this linear combination eliminates the first $\mu_{\text{old}}-1$ initial entries. The μ_{old}-th entry can be eliminated as in (2.7) by subtracting the right multiple of column $\tilde{\mathbf{S}}_{\nu_{\text{old}}}$ and updating Λ^{new} accordingly. Similarly entries $\mu_{\text{old}}+1$ up to μ can be eliminated as in (2.8) by subtracting multiples of columns $\tilde{\mathbf{S}}_{\nu_{\text{new}}-1}$ down to $\tilde{\mathbf{S}}_{\nu}$ similarly and updating Λ^{new} accordingly.

The last claim we prove by induction. In the beginning we have $\mu_{\text{old}}=0$ and $\nu=1=\mu_{\text{old}}+1$. Suppose the first discrepancy from zero we find in row μ, then columns $\mathbf{S}_1,\ldots,\mathbf{S}_\mu$ are obviously linearly independent and the FIA can jump to column $\nu_{\text{new}}=\mu+1$. The hypothesis holds under initial conditions. In general a jump to column $\nu_{\text{new}}=\nu+\mu-\mu_{\text{old}}$ is suggested. By induction we know that $\nu=\mu_{\text{old}}+1$ and therefore $\nu_{\text{new}}=\mu+1$. □

Theorem 2.2 shows how to adjust the FIA in order to run it more efficiently on syndrome matrices. In the beginning set $\Lambda^{\text{old}}=0$ and $\Lambda=1$. If the algorithm stalls in a column with polynomial Λ, then it is not necessary to consider certain subsequent columns, but they may be jumped over. In the new column it is known how many entries can be eliminated and how to update Λ efficiently. Adjusting the FIA in this way is also due to Feng and Tzeng (1991). In Algorithm 2.2 a formal description of this adjusted FIA is depicted. It includes however the generation of the error evaluator polynomial which will first be discussed in the next section. Note that only three "generations" of variables are needed.

From now on when discussing the BMA we mean more specifically the adjusted FIA. The minor difference to the original publications by Berlekamp (1968) and Massey (1969) consists in the following: Instead of using Λ^{old} and δ_{old} from column ν_{old}, row μ_{old} when updating in column ν_{new}, row μ_{old}, they use the intermediate Λ and δ from column ν, row μ_{old} whenever the discrepancy δ there is non-zero. This is however insignificant for the discussions in this paper as we will see in Section 2.4.2.

2.3.4 Generating Ω and/or Ψ Simultaneously to Λ

So far we have not mentioned how to generate the error evaluator polynomial Ω or coevaluator Ψ. Of course this can be done by simply computing $\Lambda\cdot S$, then by the key equation (2.1) they are immediately given. But it can also be done in an iterative fashion. At every stage of the adjusted FIA run on the syndrome matrix $\mathbf{S}$ generating Λ, we have an equation of the same kind as (2.1). Suppose it has reached column ν and row μ with discrepancy δ in this place, then

$$\Lambda\cdot S=\Omega-\Psi x^{\nu+\mu-2} \tag{2.10}$$

where Λ is a polynomial of degree $\leq\nu-1$, Ω one of degree $<\nu-1$, Ψ one of degree $\leq n-k-\mu$ and δ equals the constant term of $-\Psi$. In matrix form that

is

$$\begin{pmatrix} 0 & \cdots & 0 & S_0 \\ \vdots & & \vdots & \\ 0 & \iddots & S_{\nu-2} & \\ S_0 & & S_{\nu-1} & \\ \vdots & & \vdots & \\ S_{\mu-2} & \cdots & S_{\mu+\nu-3} & \\ S_{\mu-1} & \cdots & S_{\mu+\nu-2} & \\ & & \vdots & \\ \vdots & & S_{n-k-1} & \\ & & 0 & \\ & \iddots & \vdots & \\ S_{n-k-1} & 0 \cdots & 0 & \end{pmatrix} \begin{pmatrix} \Lambda_{\nu-1} \\ \vdots \\ \Lambda_1 \\ 1 \end{pmatrix} = \begin{pmatrix} \Omega_0 \\ \vdots \\ \Omega_{\nu-2} \\ 0 \\ \vdots \\ 0 \\ -\Psi_0 \\ \\ \\ \vdots \\ \\ \\ -\Psi_{n-k-\mu} \end{pmatrix} \tag{2.10'}$$

similarly to (2.3). In column $\epsilon+1$ of $\mathbf{S}$ when the discrepancy in the t'-th row has been eliminated the actual error locator is obtained and equation (2.10) becomes

$$\Lambda \cdot S = \Omega - \Psi x^{t'+\epsilon}$$

for $\nu = \epsilon+1$ and $\mu = t'+1$ with $\deg\Lambda = \epsilon$, $\deg\Omega < \epsilon$ and $\deg\Psi < t$ which by Proposition 2.1 is equivalent to the key equation (2.1). This illustrates that Ω or Ψ can be generated simultaneously to Λ in the adjusted FIA: When Λ is the current preliminary version of the error locator in column ν row μ, current Ω and Ψ polynomials can be generated by building the corresponding linear combinations of entries in the first $\nu-1$ rows and the last $n-k-\mu+1$ rows as in (2.10'). However it can be done more efficiently from the knowledge of previous polynomials Ω^{old} and Ψ^{old} similarly to the computation of Λ. The following theorem describes how Ω and Ψ can be initialized and updated iteratively in the setting of the BMA.

Theorem 2.3 *Set* $\Omega = -1$, $\Omega^{\text{new}} = 0$, $\Psi = -1$, $\Psi^{\text{new}} = -S$, $\delta = 1$, $\mu = 0$, $\mu_{\text{new}} = 1$ *and* $\nu = 2$, $\nu_{\text{new}} = 1$ *and perform the adjusted FIA on syndrome matrix* $\mathbf{S}$. *Equation (2.10) holds at every stage for the polynomials of one "generation" if the algorithm is supplemented in the following way:*

1. *When the algorithm jumps from column* ν *to* ν_{new}, *polynomials* Ω^{new} *and* Ψ^{new} *respectively are set to*

$$\Omega - \frac{\delta}{\delta_{\text{old}}}\Omega^{\text{old}} x^{\nu_{\text{new}}-\nu_{\text{old}}}, \qquad \left(\Psi - \frac{\delta}{\delta_{\text{old}}}\Psi^{\text{old}}\right)\cdot x^{-1}, \tag{2.7'}$$

2. *when a discrepancy* $\delta_{\text{new}} \neq 0$ *is found in row* μ_{new} *where* $\mu_{\text{old}} < \mu_{\text{new}} \leq \mu$, Ω^{new} *and* Ψ^{new} *respectively are updated to*

$$\Omega^{\text{new}} - \frac{\delta_{\text{new}}}{\delta}\Omega x^{\mu_{\text{new}}-\nu+1}, \qquad \left(\Psi^{\text{new}} - \frac{\delta_{\text{new}}}{\delta}\Psi\right)\cdot x^{-1} \tag{2.8'}$$

and additionally

3. *whenever a discrepancy $\delta = 0$ has been found the current Ψ is divided by x.*

Proof : We prove the claim by induction.

Note that in the beginning before anything has been eliminated

$$\Lambda \cdot S = 0 = -1 - (-1 \cdot x^0) = \Omega - \Psi x^{\nu+\mu-2},$$
$$\Lambda^{\text{new}} \cdot S = S = 0 - (-S \cdot x^0) = \Omega^{\text{new}} - \Psi^{\text{new}} x^{\nu_{\text{new}}+\mu_{\text{new}}-2}$$

where $\deg \Lambda \leq 1 = \nu - 1$, $\deg \Lambda^{\text{new}} = 0 \leq 0 = \nu_{\text{new}} - 1$, $\deg \Omega = 0 < 1 = \nu - 1$, $\deg \Omega^{\text{new}} < 0 = \nu_{\text{new}} - 1$, $\deg \Psi = 0 \leq n - k = n - k - \mu$, $\deg \Psi^{\text{new}} \leq n - k - 1 = n - k - \mu_{\text{new}}$ and δ, δ_{new} equal the constant term of $-\Psi$ and $-\Psi^{\text{new}}$ respectively.

Suppose that the algorithm reaches column ν and row μ of the syndrome matrix $\mathbf{S}$ with discrepancy $\delta \neq 0$ such that $\mu > \mu_{\text{old}}$. In other words it jumps to column $\nu_{\text{new}} = \mu + 1$, row $\mu_{\text{new}} = \mu_{\text{old}}$ where δ has to be eliminated. At this stage we have

$$\Lambda \cdot S = \Omega - \Psi x^{\nu+\mu-2} = \Omega - \Psi x^{\nu_{\text{new}}+\mu_{\text{old}}-2} = \Omega - \Psi x^{\nu_{\text{new}}+\mu_{\text{new}}-2}$$

with $\deg \Lambda \leq \nu - 1 \leq \nu_{\text{new}} - 1$, $\deg \Omega < \nu - 1 < \nu_{\text{new}} - 1$ and $\deg \Psi \leq n - k - \mu \leq n - k - \mu_{\text{new}}$.

When discrepancy δ is eliminated and μ_{new} set to $\mu_{\text{new}} + 1 = \mu_{\text{old}} + 1$ where the next discrepancy is, we get

$$\begin{aligned}
\Lambda^{\text{new}} \cdot S &= \left(\Lambda - \frac{\delta}{\delta_{\text{old}}} \Lambda^{\text{old}} x^{\nu_{\text{new}}-\nu_{\text{old}}} \right) \cdot S \\
&= \Lambda \cdot S - \frac{\delta}{\delta_{\text{old}}} \Lambda^{\text{old}} \cdot S x^{\nu_{\text{new}}-\nu_{\text{old}}} \\
&= \Omega - \Psi x^{\nu+\mu-2} - \frac{\delta}{\delta_{\text{old}}} \left(\Omega^{\text{old}} - \Psi^{\text{old}} x^{\nu_{\text{old}}+\mu_{\text{old}}-2} \right) \cdot x^{\nu_{\text{new}}-\nu_{\text{old}}} \\
&= \Omega^{\text{new}} - \left(\Psi - \frac{\delta}{\delta_{\text{old}}} \Psi^{\text{old}} \right) x^{-1} \cdot x^{\nu_{\text{new}}+\mu_{\text{old}}-1} \\
&= \Omega^{\text{new}} - \Psi^{\text{new}} x^{\nu_{\text{new}}+\mu_{\text{old}}-1} \\
&= \Omega^{\text{new}} - \Psi^{\text{new}} x^{\nu_{\text{new}}+\mu_{\text{new}}-2}.
\end{aligned}$$

Suppose now on the other hand that the algorithm has eliminated discrepancy $\delta_{\text{new}} \neq 0$ in column ν_{new}, row μ_{new} such that $\mu_{\text{old}} + 1 \leq \mu_{\text{new}} \leq \mu$. The next discrepancy is in row $\mu_{\text{new}} + 1$ and we calculate

$$\begin{aligned}
&\left(\Lambda^{\text{new}} - \frac{\delta_{\text{new}}}{\delta} \Lambda x^{\mu_{\text{new}}-\nu+1} \right) \cdot S \\
&\quad = \Lambda^{\text{new}} \cdot S - \frac{\delta_{\text{new}}}{\delta} \Lambda \cdot S x^{\mu_{\text{new}}-\nu+1} \\
&\quad = \Omega^{\text{new}} - \Psi^{\text{new}} x^{\nu_{\text{new}}+\mu_{\text{new}}-2} - \frac{\delta_{\text{new}}}{\delta} \left(\Omega - \Psi x^{\nu+\mu-2} \right) \cdot x^{\mu_{\text{new}}-\nu+1} \\
&\quad = \Omega^{\text{new}} - \frac{\delta_{\text{new}}}{\delta} \Omega x^{\mu_{\text{new}}-\nu+1} - \left(\Psi^{\text{new}} - \frac{\delta_{\text{new}}}{\delta} \Psi \right) x^{-1} x^{\nu_{\text{new}}+(\mu_{\text{new}}+1)-2}.
\end{aligned}$$

Input: syndrome matrix $\mathbf{S}$
Output: error locator and evaluator polynomial Λ and Ω

1: Set $\nu = 2$, $\mu = 0$, $\Lambda = 0$, $\Omega = -1$ and $\delta = 1$.
2: Start with column index $\nu_{\text{new}} = 1$ and row index $\mu_{\text{new}} = 1$.
3: Initialize $\Lambda^{\text{new}} = 1$ and $\Omega^{\text{new}} = 0$.
4: **repeat**
5: Calculate discrepancy

$$\delta_{\text{new}} = \sum_{j=0}^{\nu_{\text{new}}-1} \Lambda_j S_{\nu_{\text{new}}+\mu_{\text{new}}-j-2}.$$

6: **if** $\delta_{\text{new}} \neq 0$ **then**
7: **if** $\mu_{\text{new}} \leq \mu$ **then**
8: Update Λ^{new} and Ω^{new} respectively by

$$\Lambda^{\text{new}} - \frac{\delta_{\text{new}}}{\delta}\Lambda x^{\mu_{\text{new}}-\nu+1}, \qquad \Omega^{\text{new}} - \frac{\delta_{\text{new}}}{\delta}\Omega x^{\mu_{\text{new}}-\nu+1}.$$

9: **else** $\{\mu_{\text{new}} > \mu\}$
10: Set

$$\begin{aligned} &\delta_{\text{old}} = \delta, \quad \delta = \delta_{\text{new}}, && \\ &\mu_{\text{old}} = \mu, \quad \mu = \mu_{\text{new}}, && \Lambda_{\text{old}} = \Lambda, \quad \Lambda = \Lambda_{\text{new}}, \\ &\nu_{\text{old}} = \nu, \quad \nu = \nu_{\text{new}}, && \Omega_{\text{old}} = \Omega, \quad \Omega = \Omega_{\text{new}}. \end{aligned}$$

11: Set $\mu_{\text{new}} = \mu_{\text{old}}$, $\nu_{\text{new}} = \mu + 1$.
12: Set Λ^{new} and Ω^{new} to

$$\Lambda - \frac{\delta}{\delta_{\text{old}}}\Lambda^{\text{old}} x^{\nu_{\text{new}}-\nu_{\text{old}}}, \qquad \Omega - \frac{\delta}{\delta_{\text{old}}}\Omega^{\text{old}} x^{\nu_{\text{new}}-\nu_{\text{old}}}.$$

13: **end if**
14: **end if**
15: Increment μ_{new} by 1.
16: **until** $\mu_{\text{new}} > t$
17: **return** Λ^{new} and Ω^{new}

Algorithm 2.2: The Berlekamp-Massey algorithm in matrix form

The degree properties in both cases follow by careful comparison of degrees and the fact that the constant terms of Ψ and $\frac{\delta}{\delta_{\text{old}}}\Psi^{\text{old}}$, similarly of Ψ^{new} and $\frac{\delta_{\text{new}}}{\delta}\Psi$, cancel each other. Once this is established, it also follows that the constant term of $-\Psi^{\text{new}}$ equals δ.

Finally when a discrepancy $\delta = 0$ is found in column ν_{new}, row μ_{new}, it means that the current Ψ has no constant term and therefore (2.10) also holds after moving onto the row $\mu_{\text{new}} + 1$ and division of Ψ by x. □

Algorithm 2.2 gives the formal description of the adjusted version of the FIA which includes the generation of Ω as it is conventionally the error evaluator

Λ	Ω	ν	μ	δ
0	-1	2	0	1
1	0	1	1	α^{14}
1	α^{14}	2	1	α^{13}
$\alpha^{14}x+1$	α^{14}	2	2	α^{12}
$\alpha^{13}x^2+\alpha^{14}x+1$	α^{14}	3	5	α^{10}
$\alpha^{12}x^5+\alpha^{13}x^4+\alpha^{13}x^2+\alpha^{14}x+1$	$\alpha^{12}x^4+\alpha^{14}$	6	4	α^{8}
$\alpha^{12}x^5+\alpha^{4}x^4+\alpha^{12}x^3+\alpha^{14}x+1$	$\alpha^{12}x^4+\alpha^{12}x^2+\alpha^{14}$	6	6	α^{4}
$\alpha^{7}x^6+\alpha^{9}x^5+\alpha^{14}x^4+\alpha^{12}x^3+\alpha^{14}x+1$	$\alpha^{9}x^4+\alpha^{12}x^2+\alpha^{14}$	7	8	

Table 2.1: Variable values during the run of the BMA in Example 2.1

that is used in the Forney algorithm (2.2). Note that this is also reasonable in this setting as more memory space is needed to store versions of Ψ than those of Ω. Correctness of the algorithm is given by Theorem 2.2 and 2.3.

To illustrate the course of this version of the FIA an example will be given.

Example 2.1 Reed-Solomon codes are alternant codes with $m=1$ and $a=y$. Consider the $(15,1,15)$ Reed-Solomon code over $\mathbb{F}_{16}$ with primitive element α satisfying $\alpha^4=\alpha+1$ and $\mathbf{a}=\mathbf{y}=(1,\alpha,\alpha^2,\ldots,\alpha^{14})$. We assume that the zero word was sent, but $r=(0,1,1,0,0,0,1,1,1,0,0,0,0,1,0)$ received. The parity check matrix H can be set up and syndromes S_i, $1\leq i\leq 14$ calculated which results in the following syndrome matrix

$$\mathbf{S}=\begin{pmatrix} \alpha^{14} & \alpha^{13} & 0 & \alpha^{11} & \alpha^{10} & 0 & \alpha & \alpha^{7} \\ \alpha^{13} & 0 & \alpha^{11} & \alpha^{10} & 0 & \alpha & \alpha^{7} & 0 \\ 0 & \alpha^{11} & \alpha^{10} & 0 & \alpha & \alpha^{7} & 0 & \alpha^{5} \\ \alpha^{11} & \alpha^{10} & 0 & \alpha & \alpha^{7} & 0 & \alpha^{5} & \alpha^{8} \\ \alpha^{10} & 0 & \alpha & \alpha^{7} & 0 & \alpha^{5} & \alpha^{8} & 0 \\ 0 & \alpha & \alpha^{7} & 0 & \alpha^{5} & \alpha^{8} & 0 & \alpha^{4} \\ \alpha & \alpha^{7} & 0 & \alpha^{5} & \alpha^{8} & 0 & \alpha^{4} & \alpha^{2} \end{pmatrix}.$$

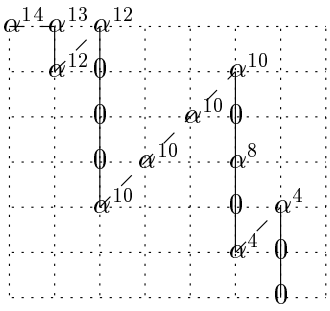
The different values of the variables Λ, Ω, ν, μ and $\delta\neq 0$ during the run of the BMA on $\mathbf{S}$ as are shown in Table 2.1 whereas the way the algorithm traverses the matrix $\mathbf{S}$ is shown in Figure 2.2(a). As a result we obtain the error locator polynomial

$$\begin{aligned}\Lambda &= \alpha^{7}x^6+\alpha^{9}x^5+\alpha^{14}x^4+\alpha^{12}x^3+\alpha^{14}x+1\\ &= (1-\alpha^{1}x)(1-\alpha^{2}x)(1-\alpha^{6}x)(1-\alpha^{7}x)(1-\alpha^{8}x)(1-\alpha^{13}x)\end{aligned}$$

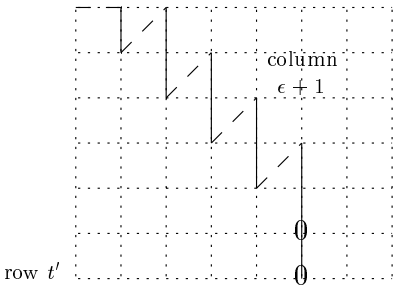
which indicates the right error locations. With the help of the error evaluator polynomial $\Omega=\alpha^9x^4+\alpha^{12}x^2+\alpha^{14}$ and the Forney Algorithm (2.2) error values can be determined.

2.3.5 Complexity of the Berlekamp-Massey Algorithm

We investigate the time complexity of the BMA. Under the condition that $\epsilon\leq t$ errors occured, our worst case assumption, depicted in Figure 2.2(b), is that



(a) Illustration of the course of the BMA in Example 2.1 including the discrepancies



(b) Illustration of the worst case situation for the BMA on a syndrome matrix with $\epsilon = 5 \leq 7 = t$ errors — all discrepancies are $\neq 0$ except those marked 0.

Figure 2.2: Courses of the BMA

$S_0 \neq 0$ and that in one column the BMA is only able to eliminate the two non-zero discrepancies in rows μ_{old} and $\mu = \mu_{\text{old}} + 1$ and stalls again in $\mu_{\text{new}} = \mu + 1$. For simplicity only multiplications and divisions will be considered. We use that the different versions of Λ which are of degree $\nu - 1$ always have constant term 1, therefore only $\nu - 1$ multiplications are needed when calculating discrepancies and updates of Λ. Updating in column $\nu_{\text{new}} = 2, \ldots, \epsilon + 1$ therefore involves

1. finding one discrepancy in column ν row μ and one in column $\nu_{\text{new}} = \nu + 1$ row μ. As $\Lambda = \Lambda^{\text{new}} = 1$ when $\nu_{\text{new}} = 2$ no multiplications are needed here, but $\nu + \nu_{\text{new}} - 2 = 2\nu - 1$ multiplications are needed for $\nu_{\text{new}} \geq 3$.
2. updating Λ^{new} once for $\nu_{\text{new}} = 2$ resulting in 1 division and 0 multiplications and twice for $\nu_{\text{new}} \geq 3$ resulting in 2 divisions and $\nu_{\text{old}} + \nu - 2 = 2\nu - 3$ multiplications.
3. updating Ω^{new} once for $\nu_{\text{new}} = 2$ takes no multiplications, updating it twice for $\nu_{\text{new}} \geq 3$ takes $\nu_{\text{old}} + \nu - 2 = 2\nu - 3$ multiplications. No divisions are necessary as those made for updating Λ^{new} can be reused.

After column $\epsilon + 1$ row ϵ has been reached and the discrepancy there eliminated, all subsequent discrepancies are 0, but need calculating. Therefore $(t - \epsilon)\epsilon$ further multiplications are carried out in the end. Adding up over all the columns gives a maximum of

$$(t - \epsilon)\epsilon + \sum_{\nu=2}^{\epsilon}(6\nu - 7) = (t - \epsilon)\epsilon + 3\epsilon(\epsilon - 1) - (\epsilon - 1) = t\epsilon + 2\epsilon^2 - 4\epsilon + 1$$

multiplications and $2\epsilon - 1$ divisions. This is the best upper bound on the number of operations known to the authors.

2.4 The Euclidean Algorithm for Decoding

2.4.1 The Original Algorithm

Now, we compare the BMA from the previous section to the EA. The ancient, extended Euclidean algorithm performed on two elements a, b of a Euclidean ring (like for example any polynomial ring in one variable) calculates repeatedly elements f, g, h of the same ring such that

$$h = fa + gb$$

until $h = \gcd(a, b)$. The key equation (2.1) reminds of the above as we can write

$$\Omega = \Lambda \cdot S + \Psi \cdot x^{n-k}$$

except that Ω is in general not the greatest common divisor of x^{n-k} and S. However Sugiyama et al. (1975) proved that by applying the extended Euclidean algorithm to polynomials x^{n-k} and S, terminating and normalizing when certain degree conditions are satisfied, the error locator, evaluator and coevaluator polynomials Λ, Ω and Ψ are obtained. The resulting EA is shown in Algorithm 2.3. Note that after every iteration we have the equality

$$\begin{pmatrix} \rho_{\text{old}} \\ \rho \end{pmatrix} = A \begin{pmatrix} x^{n-k} \\ S \end{pmatrix}. \tag{2.11}$$

The determinant of A is $(-1)^p$ for some integer p and therefore

$$\begin{pmatrix} x^{n-k} \\ S \end{pmatrix} = A^{-1} \begin{pmatrix} \rho_{\text{old}} \\ \rho \end{pmatrix} = (-1)^p \begin{pmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{pmatrix} \begin{pmatrix} \rho_{\text{old}} \\ \rho \end{pmatrix}. \tag{2.11'}$$

We know that $\deg \rho_{\text{old}} > \deg \rho$ and by the way that A is updated in Step 4 of the algorithm it follows that $\deg a_{21} > \deg a_{11}$ and $\deg a_{22} > \deg a_{12}$. Terminating as soon as $\deg \rho < t$, means that $\deg \rho_{\text{old}} \geq t'$ and therefore by (2.11') $\deg a_{22} = \deg x^{n-k} - \deg \rho_{\text{old}} \leq n - k - t' = t$ and $\deg a_{21} = \deg S - \deg \rho_{\text{old}} \leq n - k - 1 - t' < n - k - t' = t$ and a_{22}, ρ, a_{21} fulfill

$$\rho = a_{21} x^{n-k} + a_{22} S.$$

by (2.11) and therefore the key equation (2.1). Further, $\det A = a_{11}a_{22} - a_{12}a_{21} = (-1)^p$ implies that $\gcd(a_{22}, a_{21}) = 1$ and therefore with Theorem 2.1 the error locator polynomial is $\Lambda = a_{22}/a_{22}(0)$, the error evaluator $\Omega = \rho/a_{22}(0)$ and the error coevaluator is $\Psi = a_{21}/a_{22}(0)$.

2.4.2 A First Adjustment

At first sight the EA seems to be different from the BMA, since intermediate results do not coincide. Two immediately obvious reasons why they cannot coincide in general, are

1. that the EA starts its arithmetic computations with $S_{n-k-1}, S_{n-k-2}, \ldots$ when doing the first polynomial divisions whereas the BMA starts with $S_1, S_2, \ldots$, and

Input: syndrome polynomial $S \neq 0$, integer t
Output: error locator, evaluator and coevaluator polynomials Λ, Ω, Ψ

1: Initialize $\rho_{\text{old}} = x^{n-k}$, $\rho = S$ and

$$A = (a_{ij})_{1 \leq i,j \leq 2} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

2: **repeat**
3: Compute quotient q of ρ_{old} by ρ and remainder ρ_{new} such that

$$\rho_{\text{old}} = q\rho + \rho_{\text{new}}, \quad \deg \rho_{\text{new}} < \deg \rho.$$

4: Update A by

$$\begin{pmatrix} 0 & 1 \\ 1 & -q \end{pmatrix} A.$$

Set $\rho_{\text{old}} = \rho, \quad \rho = \rho_{\text{new}}$.
5: **until** $\deg \rho < t$
6: **return** $a_{22}/a_{22}(0)$, $\rho/a_{22}(0)$, $a_{21}/a_{22}(0)$

Algorithm 2.3: The original Euclidean algorithm

2. that the preliminary version of the error locator polynomial a_{22} is not kept to have constant term 1 at all times.

In the following we denote the reciprocal of a polynomial $f \in \mathbb{F}[x]$ by $f^{\text{rec}} = x^{\deg f} f(x^{-1})$. Further we define the *reversed syndrome polynomial*

$$\bar{S} = x^{n-k-1} S(x^{-1}) = S_0 x^{n-k-1} + S_1 x^{n-k-2} + \cdots + S_{n-k-1}.$$

Note that $\bar{S}$ is generally not equal to S^{rec}. We investigate running the EA on $\bar{S}$ instead. When doing so it cannot be expected that the error locator polynomial turns up as a normalized version of a_{22}, but rather as something like the reciprocal of it. Correspondingly, instead of normalizing at the end, a_{22} should be kept monic throughout the course of the EA. The algorithm we propose is shown in Algorithm 2.4. Note in particular the equality

$$\begin{pmatrix} 0 & 1 \\ Q & -q \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -q_0 & 1 \end{pmatrix} \cdots \begin{pmatrix} 1 & 0 \\ -q_{d-1}x^{d-1} & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ Q & x^d \end{pmatrix}$$

and the similarity to the algorithm in Algorithm 2.3. Further, recall polynomial division: Whenever a new term of q is obtained an intermediate ρ_{new} is calculated that has degree $\geq \deg \rho$ until the division has reached its end. By these considerations, the equation corresponding to (2.11)

$$\begin{pmatrix} \rho \\ \rho_{\text{new}} \end{pmatrix} = A \begin{pmatrix} x^{n-k} \\ \bar{S} \end{pmatrix} \tag{2.12}$$

holds not only after each polynomial division where $\deg \rho_{\text{new}} < \deg \rho$, but also when one polynomial division is not yet complete and an intermediate ρ_{new} has

Input: reversed syndrome polynomial $\bar{S} \neq 0$, integer t
Output: error locator, evaluator and coevaluator polynomials Λ, Ω, Ψ

1: Initialize $\rho_{\text{old}} = x^{n-k}$, $\rho = \bar{S}$ and

$$A = (a_{ij})_{1 \leq i,j \leq 2} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

2: **repeat**
3: Compute constant Q, quotient $q = -x^d + q_{d-1}x^{d-1} + \cdots + q_0$ of ρ_{old} by ρ and remainder ρ_{new} such that

$$Q\rho_{\text{old}} = q\rho + \rho_{\text{new}}, \quad \deg \rho_{\text{new}} < \deg \rho.$$

4: Update A simultaneously to obtaining terms of q by

$$\begin{pmatrix} 1 & 0 \\ -q_0 & 1 \end{pmatrix} \cdots \begin{pmatrix} 1 & 0 \\ -q_{d-1}x^{d-1} & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ Q & x^d \end{pmatrix} A.$$

Set $\rho_{\text{old}} = \rho, \quad \rho = \rho_{\text{new}}$.
5: **until** $\deg \rho < t$
6: **return** a_{22}^{rec}, $-x^{\deg a_{22}-1} a_{21}(x^{-1})$, $-\rho^{\text{rec}}$

Algorithm 2.4: The adjusted Euclidean algorithm - version 1

been calculated with $\deg \rho_{\text{new}} \geq \deg \rho$. The coefficient of the term of largest degree in a general polynomial $f \in \mathbb{F}[x]$ is called the *leading coefficient* of f, abbreviated $\operatorname{lc} f$. The reason why we choose $\operatorname{lc} q = -1$ is that $-q$ and a_{22} then are always monic.

The following theorem draws the parallels between the BMA and the new version of the EA. Thereby it also proves correctness of the adjusted EA.

Theorem 2.4 *Suppose both the BMA from Algorithm 2.2 including the generation of Ψ as in Theorem 2.3 and the first version of the adjusted EA from Algorithm 2.4 are run on the non-trivial syndromes of the same error pattern. After every update of Λ^{new}, Ω^{new}, Ψ^{new}, ρ_{new}, a_{22}^{new} and a_{21}^{new} and setting μ_{new} to the index of the next row with discrepancy $\delta_{\text{new}} \neq 0$, we have*

1. $a_{22}^{\text{new rec}} = \Lambda^{\text{new}}$ *and* $\deg a_{22}^{\text{new}} = \nu_{\text{new}} - 1$ *for* $a_{22}^{\text{new}} \neq 0$,
2. $-x^{\deg a_{22}^{\text{new}}-1} a_{21}^{\text{new}}(x^{-1}) = \Omega^{\text{new}}$ *for* $a_{22}^{\text{new}} \neq 0$ *and* $\deg a_{21}^{\text{new}} < \nu_{\text{new}} - 1$,
3. $-\rho_{\text{new}}^{\text{rec}} = \Psi^{\text{new}}$ *and* $\deg \rho_{\text{new}} = n - k - \mu_{\text{new}}$.

In particular $\operatorname{lc} \rho_{\text{new}} = \delta_{\text{new}}$. *Furthermore, there is a one to one correspondence between column shifts in the BMA and the start of a new polynomial division in the adjusted EA as well as a one to one correspondence between every update of Λ and Ω within one column in the BMA and the number of non-trivial updates of a_{22} and a_{21} within one polynomial division in the adjusted EA.*

Lemma 2.1 *Let $f, g, h \in \mathbb{F}[x] \setminus \{0\}$, $\lambda, \mu \in \mathbb{F}^*$, $i \in \mathbb{N}_0$ such that $f = \lambda g + \mu x^i h$.*

1. *If $\deg f = \deg g$, then $f^{\mathrm{rec}} = \lambda g^{\mathrm{rec}} + \mu x^{\deg g - (i + \deg h)} h^{\mathrm{rec}}$.*
2. *If $\deg f = i + \deg h$, then $f^{\mathrm{rec}} = \lambda x^{i + \deg h - \deg g} g^{\mathrm{rec}} + \mu h^{\mathrm{rec}}$.*

Proof: 1.

$$\begin{aligned} f^{\mathrm{rec}} &= x^{\deg f} f(x^{-1}) \\ &= x^{\deg g} \left(\lambda g(x^{-1}) + \mu x^{-i} h(x^{-1})\right) \\ &= x^{\deg g} \left(\lambda x^{-\deg g} g^{\mathrm{rec}} + \mu x^{-(i+\deg h)} h^{\mathrm{rec}}\right) \\ &= \lambda g^{\mathrm{rec}} + \mu x^{\deg g - (i + \deg h)} h^{\mathrm{rec}} \end{aligned}$$

2.

$$\begin{aligned} f^{\mathrm{rec}} &= x^{\deg f} f(x^{-1}) \\ &= x^{i + \deg h} \left(\lambda g(x^{-1}) + \mu x^{-i} h(x^{-1})\right) \\ &= x^{i + \deg h} \left(\lambda x^{-\deg g} g^{\mathrm{rec}} + \mu x^{-(i+\deg h)} h^{\mathrm{rec}}\right) \\ &= \lambda x^{i + \deg h - \deg g} g^{\mathrm{rec}} + \mu h^{\mathrm{rec}} \end{aligned}$$

□

Proof of Theorem 2.4: First a remark on notation: Suppose A from the adjusted EA has just been updated, then $a_{11}^{\mathrm{new}} = a_{21}$ and $a_{12}^{\mathrm{new}} = a_{22}$ where a_{21}, a_{22} denote entries from the previous polynomial division. In this way we can restrict our considerations to three "generations" of a_{21} and a_{22} — old, current and new. Note that $\deg a_{21} < \deg a_{21}^{\mathrm{new}}$ and $\deg a_{22} < \deg a_{22}^{\mathrm{new}}$.

We prove the claim by induction on the steps of the two algorithms.

In the beginning, we have

1. $a_{22}^{\mathrm{old\,rec}} = 0 = \Lambda^{\mathrm{old}}$, $a_{22}^{\mathrm{rec}} = 1 = \Lambda$, and $\deg a_{22} = 0 = \nu - 1$,
2. $\deg a_{21}^{\mathrm{old}} = 0 < \nu_{\mathrm{old}} - 1$ and $-x^{\deg a_{22} - 1} a_{21}(x^{-1}) = 0 = \Omega$, $\deg a_{21} < \nu - 1$,
3. $-\rho_{\mathrm{old}}^{\mathrm{rec}} = -1 = \Psi^{\mathrm{old}}$ and $\deg \rho_{\mathrm{old}} = n - k = n - k - \mu_{\mathrm{old}}$, $-\rho^{\mathrm{rec}} = -Sx^{-(\mu-1)} = \Psi$ and $\deg \rho = n - k - \mu$.

In particular $\mathrm{lc}\,\rho_{\mathrm{old}} = 1 = \delta_{\mathrm{old}}$ and $\mathrm{lc}\,\rho = S_{\mu-1} = \delta$. The claim of Theorem 2.4 holds for the first two generations, and we may assume in the investigation of a step that the theorem holds for previous steps of the two algorithms.

Suppose the BMA finds a non-zero discrepancy. We distinguish between two cases.

The first case is that this discrepancy $\delta \neq 0$ in column ν, row μ cannot be eliminated and a column change is forced such that $\nu_{\mathrm{new}} = \mu + 1$, $\mu_{\mathrm{new}} = \mu_{\mathrm{old}}$ and

$$\Lambda^{\mathrm{new}} = \Lambda - \frac{\delta}{\delta_{\mathrm{old}}} \Lambda^{\mathrm{old}} x^{\nu_{\mathrm{new}} - \nu_{\mathrm{old}}}.$$

This implies $\mu > \mu_{\text{old}}$ and $\mu \leq t'$, thus $t = n - k - t' \leq n - k - \mu = \deg \rho < \deg \rho_{\text{old}} = n - k - \mu_{\text{old}}$ such that the adjusted EA starts a new polynomial division of ρ_{old} by ρ. It sets $d = \deg \rho_{\text{old}} - \deg \rho$ and $Q = -\text{lc}\,\rho/\text{lc}\,\rho_{\text{old}}$ such that

$$a_{22}^{\text{new}} = -\frac{\text{lc}\,\rho}{\text{lc}\,\rho_{\text{old}}} a_{22}^{\text{old}} + x^{\deg \rho_{\text{old}} - \deg \rho} a_{22}. \tag{2.13}$$

Note that $a_{22}^{\text{new}} = x^{\mu}$ during the first polynomial division and therefore $a_{22}^{\text{new rec}} = 1 = \Lambda$. Subsequently, when $a_{22}^{\text{old}} \neq 0$ we can reciprocate the expression (2.13) with the help of 2 in Lemma 2.1

$$a_{22}^{\text{new rec}} = a_{22}^{\text{rec}} - \frac{\text{lc}\,\rho}{\text{lc}\,\rho_{\text{old}}} x^{\deg \rho_{\text{old}} - \deg \rho + \deg a_{22} - \deg a_{22}^{\text{old}}} a_{22}^{\text{old rec}}.$$

But

$$a_{22}^{\text{old rec}} = \Lambda^{\text{old}}, \quad a_{22}^{\text{rec}} = \Lambda, \quad \frac{\text{lc}\,\rho}{\text{lc}\,\rho_{\text{old}}} = \frac{\delta}{\delta_{\text{old}}}$$

and

$$\begin{aligned} \deg \rho_{\text{old}} - \deg\, \rho + \deg a_{22} - \deg a_{22}^{\text{old}} & \\ &= n - k - \mu_{\text{old}} - (n - k - \mu) + \nu - 1 - \nu_{\text{old}} + 1 \\ &= \mu - \mu_{\text{old}} + \nu - \nu_{\text{old}} = \nu_{\text{new}} - \nu_{\text{old}} \end{aligned}$$

by induction. Therefore we get

$$a_{22}^{\text{new rec}} = \Lambda - \frac{\delta}{\delta_{\text{old}}} \Lambda^{\text{old}} x^{\nu_{\text{new}} - \nu_{\text{old}}} = \Lambda^{\text{new}}$$

as in (2.7). As $a_{22}^{\text{new}} = x^{\mu}$ during the first polynomial division we obtain $\deg a_{22}^{\text{new}} = \nu_{\text{new}} - 1$. Subsequently, when $a_{22}^{\text{old}} \neq 0$ we have $\deg a_{22}^{\text{old}} = \nu_{\text{old}} - 1 < \nu - 1 = \deg a_{22}$, $\deg \rho_{\text{old}} = n - k - \mu_{\text{old}}$ and $\deg \rho = n - k - \mu$ by induction and therefore

$$\begin{aligned} \deg a_{22}^{\text{new}} &= \deg a_{22} + \deg \rho_{\text{old}} - \deg \rho \\ &= \nu - 1 + n - k - \mu_{\text{old}} - (n - k - \mu) \\ &= \nu_{\text{new}} - 1. \end{aligned}$$

The second case is that this discrepancy $\delta_{\text{new}} \neq 0$ in column ν_{new}, row μ_{new} can be eliminated and Λ^{new} is updated to

$$\Lambda^{\text{new}} - \frac{\delta_{\text{new}}}{\delta} \Lambda x^{\mu_{\text{new}} - \nu + 1}$$

without column shift. I.e. we have $\mu_{\text{old}} < \mu_{\text{new}} \leq \mu$ and therefore $\deg \rho_{\text{new}} = n - k - \mu_{\text{new}} \geq n - k - \mu = \deg \rho$ such that the adjusted EA continues the polynomial division ρ_{old} by ρ. The next term of the quotient polynomial q that is calculated is therefore $\text{lc}\,\rho_{\text{new}}/\text{lc}\,\rho x^{\deg \rho_{\text{new}} - \deg \rho}$ and a_{22}^{new} is updated to

$$-\frac{\text{lc}\,\rho_{\text{new}}}{\text{lc}\,\rho} x^{\deg \rho_{\text{new}} - \deg \rho} a_{22} + a_{22}^{\text{new}}. \tag{2.13'}$$

Since $\deg a_{22}^{\text{new}} = \nu_{\text{new}} - 1 \geq \nu - 1 + \nu_{\text{new}} - \nu = \nu - 1 + n - k - \mu_{\text{new}} - (n-k-\mu) = \deg x^{\deg \rho_{\text{new}} - \deg \rho} a_{22}$, we reciprocating with 1 in Lemma 2.1 and $a_{22}^{\text{new rec}}$ is given as

$$a_{22}^{\text{new rec}} - \frac{\operatorname{lc} \rho_{\text{new}}}{\operatorname{lc} \rho} x^{\deg a_{22}^{\text{new}} - \deg \rho_{\text{new}} + \deg \rho - \deg a_{22}} a_{22}^{\text{rec}}.$$

But

$$a_{22}^{\text{rec}} = \Lambda, \quad a_{22}^{\text{new rec}} = \Lambda^{\text{new}}, \quad \frac{\operatorname{lc} \rho_{\text{new}}}{\operatorname{lc} \rho} = \frac{\delta_{\text{new}}}{\delta}$$

and

$$\begin{aligned} \deg a_{22}^{\text{new}} - \deg \rho_{\text{new}} + \deg \rho - \deg a_{22} & \\ &= \nu_{\text{new}} - 1 - (n - k - \mu_{\text{new}}) + n - k - \mu - \nu + 1 \\ &= \mu_{\text{new}} - \nu + 1 \end{aligned}$$

by induction. Therefore we get

$$a_{22}^{\text{new rec}} = \Lambda^{\text{new}} - \frac{\delta_{\text{new}}}{\delta} x^{\mu_{\text{new}} - \nu + 1} \Lambda$$

as in (2.8). Also by the above degree calculation for the reciprocation it follows that the degree of the updated a_{22}^{new} is also $\nu_{\text{new}} - 1$. Claim 1 is proven.

By the above we have also shown that there is a one to one correspondence between column shifts in the BMA and the start of a new polynomial division in the adjusted EA as well as a one to one correspondence between updates of the polynomials within one column in the BMA and the number of non-trivial updates of polynomials within one polynomial division in the adjusted EA.

Claim 2 and 3 are left to prove. In both cases whether a new polynomial division is started or continued, a polynomial

$$\rho_{\text{new}} = a_{21}^{\text{new}} x^{n-k} + a_{22}^{\text{new}} \bar{S} \tag{2.14}$$

is obtained by (2.12) and we rewrite

$$a_{22}^{\text{new}} \bar{S} = \rho_{\text{new}} - a_{21}^{\text{new}} x^{n-k} \tag{2.14'}$$

where the right hand side separates terms of the product $a_{22}^{\text{new}} \bar{S}$ into those of degree $< n - k$ and of degree $\geq n - k$. We calculate

$$\begin{aligned} \Lambda S = a_{22}^{\text{new rec}} S &= x^{\deg a_{22}^{\text{new}} + n - k - 1} a_{22}^{\text{new}}(x^{-1}) \bar{S}(x^{-1}) \\ &\overset{(2.14')}{=} x^{\deg a_{22}^{\text{new}} + n - k - 1} \left(\rho_{\text{new}}(x^{-1}) - a_{21}^{\text{new}}(x^{-1}) x^{-(n-k)} \right) \\ &= -x^{\deg a_{22}^{\text{new}} - 1} a_{21}^{\text{new}}(x^{-1}) + x^{\nu_{\text{new}} + \mu_{\text{new}} - 2} \rho_{\text{new}}^{\text{rec}} \end{aligned}$$

where the right hand side again separates terms into those of degree $\leq \nu_{\text{new}} - 2$ and of degree $> \nu_{\text{new}} - 2$. By (2.10) the equalities $-x^{\deg a_{22}^{\text{new}} - 1} a_{21}^{\text{new}}(x^{-1}) = \Omega^{\text{new}}$ and $-\rho_{\text{new}}^{\text{rec}} = \Psi^{\text{new}}$ are obtained. Further (2.14') indicates that $\deg a_{22}^{\text{new}} \bar{S} = \deg a_{21}^{\text{new}} x^{n-k}$ and therefore claim 2 follows by

$$\deg a_{21}^{\text{new}} \leq \nu_{\text{new}} - 1 + n - k - 1 - (n - k) < \nu_{\text{new}} - 1.$$

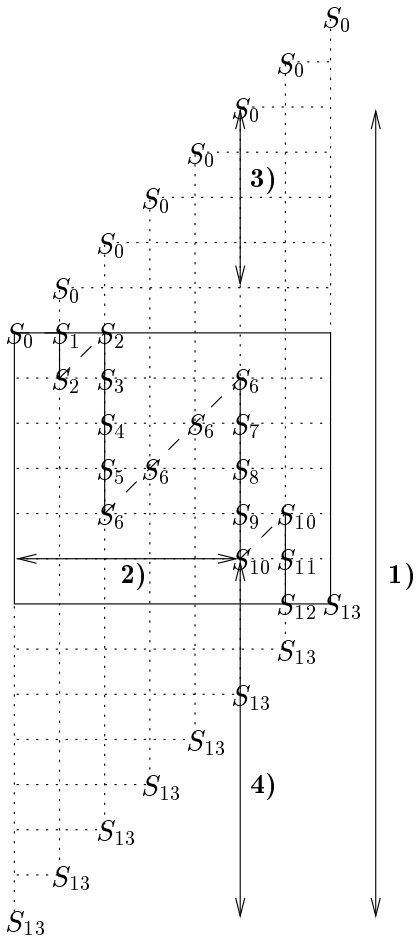


Figure 2.3: Illustration of Theorem 2.4 through Example 2.2: A matrix similar to the one in (2.3) is depicted. The solid box sets the syndrome matrix apart from the rest. When the BMA stalls in column 6, row 6 of $\mathbf{S}$ the number of coefficients in ΛS and $a_{22}\bar{S}$ corresponds to the length of the arrow marked **1)**, the number of coefficients of Λ and a_{22} corresponds to the length of the arrow marked **2)**, the number of coefficients of Ω and a_{21} corresponds to the length of the one marked **3)** and the number of coefficients of ρ is given by the length of the arrow marked **4)**. Note that some initial coefficients might be trivial and therefore it is avoided to talk about the degree of the polynomials in question.

We have already seen that $-\rho^{\text{rec}}_{\text{new}} = \Psi^{\text{new}}$. Finally let μ_{new} be the row index of the next discrepancy $\delta_{\text{new}} \neq 0$. By comparison of the coefficients of ΛS and $a^{\text{new}}_{22}\bar{S}$ from (2.14') where $a^{\text{new rec}}_{22} = \Lambda$ it can be seen that $\delta_{\text{new}} = \text{lc}\,\rho_{\text{new}}$. Claim 3 is also proven.

In particular $\delta_{\text{new}} = \text{lc}\,\rho_{\text{new}}$ by the properties of (2.10) and the BMA and the EA stop at the same time. □

Example 2.2 Let us reconsider the code and the received word from Example 2.1 and run the adjusted EA on the corresponding reversed syndrome polynomial $\bar{S}$ of degree 13. The different values of the variables a_{22}, a_{21}, ρ, Q and the terms $q_i x^i$ of the quotient polynomial q are given in Table 2.2. As expected by Theorem 2.4 the 4 column shifts illustrated in the diagram in Figure

a_{22}	a_{21}
0	1
1	0
x	α^{14}
$x+\alpha^{14}$	α^{14}
$x^2+\alpha^{14}x+\alpha^{13}$	$\alpha^{14}x$
$x^5+\alpha^{14}x^4+\alpha^{13}x^3+\alpha^{13}x+\alpha^{12}$	$\alpha^{14}x^4+\alpha^{12}$
$x^5+\alpha^{14}x^4+\alpha^{12}x^2+\alpha^4x+\alpha^{12}$	$\alpha^{14}x^4+\alpha^{12}x^2+\alpha^{12}$
$x^6+\alpha^{14}x^5+\alpha^{12}x^3+\alpha^{14}x^2+\alpha^9x+\alpha^7$	$\alpha^{14}x^5+\alpha^{12}x^3+\alpha^9x$

ρ	Q	q_ix^i
x^{14}		
$\bar{S}=\alpha^{14}x^{13}+\cdots$	α^{14}	x
$\alpha^{13}x^{13}+\cdots$		α^{14}
$\alpha^{12}x^{12}+\cdots$	α^{13}	x
$\alpha^{10}x^9+\cdots$	α^{13}	x^3
$\alpha^8x^{10}+\cdots$		$\alpha^{13}x$
$\alpha^4x^8+\cdots$	α^9	x
$\alpha^{14}x^4+\cdots$		

Table 2.2: Variable values during the run of the EA in Example 2.2

2.2(a) in the BMA, correspond to 4 polynomial divisions in the adjusted EA. Furthermore the additional updates in columns 2 and 6 correspond to the the calculation of a further term of q in the first and the third polynomial division. Comparison with the coefficients of the polynomials in the table of Figure 2.1 shows that at every point where $a_{22}^{\text{new}} \neq 0$, we have the equalities $a_{22}^{\text{rec}} = \Lambda$, $-x^{\deg a_{22}-1}a_{21}(x^{-1}) = \Omega$ and $\operatorname{lc}\rho = \delta$.

In order to understand intuitively what is going on in the two algorithms compare with Figure 2.3. Suppose the BMA finds discrepancy $\delta \neq 0$ in column ν, row μ. By (2.10) we have $\Lambda S = \Omega - \Psi x^{\nu+\mu-2}$. Its coefficients are the entries in the obvious order of the product

$$\begin{pmatrix} 0 & \cdots & 0 & S_0 \\ \vdots & \iddots & \iddots & \\ 0 & & & \vdots \\ S_0 & & & \\ & & & S_{n-k-1} \\ \vdots & & \iddots & 0 \\ & & \iddots & \vdots \\ S_{n-k-1} & 0 & \cdots & 0 \end{pmatrix} \begin{pmatrix} \Lambda_{\nu-1} \\ \vdots \\ \Lambda_1 \\ 1 \end{pmatrix},$$

where the BMA has established that entries $\nu, \ldots, \nu+\mu-2$ are 0. In other words coefficients of Ω occupy entries $1, \ldots, \nu-1$, entry $\nu+\mu-1$ is a non-zero discrepancy δ and coefficients of $-\Psi$ occupy all entries $\nu+\mu-1, \ldots, n-k+\nu-1$.

Having carried out the EA to this point, the product $a_{22}\bar{S} = \rho - a_{21}x^{n-k}$ has the same coefficients, only in reversed order by Theorem 2.4. The polynomial $-a_{21}$ corresponds to Ω and therefore $\deg a_{21} < \nu - 1$ and similarly $-\rho$ corresponds to Ψ, thus $\mathrm{lc}\,\rho = \delta$ and $\deg \rho = n - k - \mu$.

If we also want to compensate in the EA for the minor difference between the FIA and the original BMA as mentioned at the end of Section 2.3.3, then when dividing ρ_{old} by ρ and coming across a version of ρ_{new} that has the same degree as ρ, the polynomials ρ with corresponding a_{22} and a_{21} should be replaced by that ρ_{new} and corresponding versions of a_{22}^{new} and a_{21}^{new} for the next polynomial division. By Theorem 2.4, that corresponds exactly to using versions Λ, Ω, Ψ and $\delta \neq 0$ from column ν, row μ_{old} for updating in column ν_{new}, row μ_{old} in the course of the FIA.

2.4.3 Complexity of the Adjusted Euclidean Algorithm

With the first adjustment we have obtained the desired result in so far as the BMA and the EA suddenly, without having changed much, resemble each other quite a lot. Updating Λ and Ω in the one corresponds exactly to updating the matrix A in the other. However doing the polynomial division ρ_{old} by ρ seems to be more involved than simply computing discrepancies. It corresponds to computing Ψ additionally in the BMA which is not necessary.

We investigate the time complexity of the adjusted EA. Under the condition that $\epsilon \leq t$ errors have occurred, our worst case assumption is that $S_0 \neq 0$ as well as that in each division $\deg q = 1$ and q's constant term is not trivial. In other words the degrees of the remainder polynomials only fall by 1 in each step. This corresponds exactly to the worst case assumption described in Section 2.3.5 and Figure 2.2(b). For simplicity only multiplications and divisions of field elements will be considered. One polynomial division therefore involves

1. 1 division during the first polynomial division, subsequently maximally 2 divisions and $\deg \rho_{\text{old}} + \deg \rho = 2(n-k) - 2\nu + 1$ multiplications to calculate quotient q.
2. updating a_{22}^{new} twice resulting in no multiplications during the first division and maximally $\deg a_{22}^{\text{old}} + \deg a_{22} = 2\nu - 3$ multiplications subsequently and
3. updating a_{21}^{new} resulting in no multiplications during the first division and maximally $\deg a_{21}^{\text{old}} + \deg a_{21} + 2 \leq \deg a_{22}^{\text{old}} + \deg a_{22} = 2\nu - 3$ multiplications subsequently.

After ϵ polynomial divisions the final result will be obtained and adding up over all of them gives a maximum of

$$\begin{aligned} 2(n-k) - 1 + \sum_{\nu=2}^{\epsilon} & 2(n-k) - 2\nu + 1 + 4\nu - 6 \\ &= 2(n-k) - 1 + (\epsilon - 1)\big(2(n-k) - 3\big) + \epsilon(\epsilon - 1) \\ &= 4(n-k)\epsilon + \epsilon^2 - 4\epsilon + 2 \end{aligned}$$

multiplications and $2\epsilon - 1$ divisions which is strictly larger than the complexity of the BMA. Note that the complexity of the original EA in the worst case will be the same as the one of the adjusted version and that the one of the BMA would also be this large if one included the error coevaluator Ψ in the calculations as in (2.7'), (2.8').

2.4.4 A Second and Final Adjustment

It might seem impossible to get around the computations of versions of ρ in the EA. But they correspond exactly to the computation of Ψ in the BMA where they certainly can be avoided. The observations of the previous sections enables us to see how the first version of the adjusted EA can be made more efficient and resemble the BMA exactly: the remainder polynomials ρ of which several versions are calculated during one polynomial division consists of mostly superfluous information. Already in the proof of Theorem 2.4 it could be seen that only their leading coefficients are needed to compute Q and coefficients of the quotients q which are essential for updating the matrix A. The following proposition restates this observation and shows how the leading coefficients can be calculated efficiently.

Proposition 2.2 *1. When the adjusted EA starts a new polynomial division of ρ_{old} by ρ and calculates Q and q with $\operatorname{lc} q = -1$, then $Q = -\operatorname{lc}\rho/\operatorname{lc}\rho_{\text{old}}$ and the leading term of q is $-x^{\deg\rho_{\text{old}}-\deg\rho}$. The first non-zero coefficient of $x^{\deg\rho_{\text{old}}-1}, x^{\deg\rho_{\text{old}}-2}, \ldots$ in the product $a_{22}^{\text{new}}\bar{S}$ gives $\operatorname{lc}\rho_{\text{new}}$ and the degree of the correponding term gives $\deg\rho_{\text{new}}$.*

2. Within one polynomial division of the adjusted EA where an intermediate ρ_{new} has been calculated with $\deg\rho \le \deg\rho_{\text{new}} < \deg\rho_{\text{old}}$, the next term of q is calculated to be $\operatorname{lc}\rho/\operatorname{lc}\rho_{\text{new}} x^{\deg\rho_{\text{new}}-\deg\rho}$. The first non-zero coefficient of $x^{\deg\rho_{\text{new}}-1}, x^{\deg\rho_{\text{new}}-2}, \ldots$ in the product $a_{22}^{\text{new}}\bar{S}$ gives the leading coefficient of the next version of ρ_{new} and the degree of the correponding term gives its degree.

Proof: 1. Follows immediately from the fact that $Q\rho_{\text{old}} = -x^d\rho + \rho_{\text{new}}$ with $\deg\rho_{\text{new}} < \deg\rho_{\text{old}}$ should be obtained. From equation (2.12) we obtain $a_{22}^{\text{new}}\bar{S} = \rho_{\text{new}} - a_{21}^{\text{new}}x^{n-k}$, where the right hand side separates terms of the product $a_{22}^{\text{new}}\bar{S}$ into those of degree $< n-k$ and of degree $\ge n-k$. Since $\deg\rho_{\text{new}} < \deg\rho_{\text{old}}$ the leading coefficient and degree are found as described above.

2. At this stage we have $\deg\rho \le \deg\rho_{\text{new}} < \deg\rho_{\text{old}}$ which means that we divide ρ_{new} by ρ. Therefore the next term of q is $\operatorname{lc}\rho/\operatorname{lc}\rho_{\text{new}} x^{\deg\rho_{\text{new}}-\deg\rho}$. Again by equation (2.12) we have the equality $a_{22}^{\text{new}}\bar{S} = \rho_{\text{new}} - a_{21}^{\text{new}}x^{n-k}$, where the right hand side separates terms of the product $a_{22}^{\text{new}}\bar{S}$ into those of degree $< n-k$ and of degree $\ge n-k$. Since ρ_{new} is divided by ρ, the new version of ρ_{new} is of lower degree and the leading coefficient and degree are found as described above.

□

Input: reversed syndrome polynomial $\bar{S} \neq 0$, integer t
Output: error locator and evaluator polynomial Λ and Ω

1: Initialize $\deg \rho_{\text{old}} = n - k$, $\operatorname{lc} \rho_{\text{old}} = 1$, $\deg \rho = \deg \bar{S}$, $\operatorname{lc} \rho = \operatorname{lc} \bar{S}$ and

$$A = (a_{ij})_{1 \leq i,j \leq 2} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

2: **repeat**
3: Update A by

$$\begin{pmatrix} 0 & 1 \\ -\frac{\operatorname{lc} \rho}{\operatorname{lc} \rho_{\text{old}}} & x^{\deg \rho_{\text{old}} - \deg \rho} \end{pmatrix} A.$$

4: Compute coefficients $x^{\deg \rho_{\text{old}} - 1}, x^{\deg \rho_{\text{old}} - 2}, \ldots$ in $a_{22}^{\text{new}} \bar{S}$ until a non-zero one is found and $\deg \rho_{\text{new}}$, $\operatorname{lc} \rho_{\text{new}}$ obtained or until $\deg \rho_{\text{new}} < t$ has been established.
5: **while** $\deg \rho_{\text{new}} \geq \deg \rho$ **do**
6: Update A by

$$\begin{pmatrix} 1 & 0 \\ -\frac{\operatorname{lc} \rho_{\text{new}}}{\operatorname{lc} \rho} x^{\deg \rho_{\text{new}} - \deg \rho} & 1 \end{pmatrix} A.$$

7: Compute coefficients $x^{\deg \rho_{\text{new}} - 1}, x^{\deg \rho_{\text{new}} - 1}, \ldots$ in $a_{22}^{\text{new}} \bar{S}$ until a non-zero one is found and $\deg \rho_{\text{new}}$, $\operatorname{lc} \rho_{\text{new}}$ obtained or until $\deg \rho_{\text{new}} < t$ has been established.
8: **end while**
9: **until** $\deg \rho_{\text{new}} < t$
10: **return** a_{22}^{rec} and $-x^{\deg a_{22} - 1} a_{21}(x^{-1})$

Algorithm 2.5: The adjusted Euclidean algorithm - version 2

Proposition 2.2 justifies the second adjustment of the EA in Algorithm 2.5.

Note that the computation of the necessary coefficients of $a_{22}^{\text{new}} \bar{S}$ corresponds exactly to the computation of discrepancies — compare with Theorem 2.4. By the discussions in Sections 2.3.5, 2.4.3 and Theorem 2.4 the complexity of the BMA and the one of this second adjusted version of the EA are equal. The superfluous computation of the entire remainder polynomials has been replaced by discrepancy calculation and the two algorithms do essentially the same computations.

2.5 Discussion

In this paper we have chosen to alter the EA to perform the same operations as the BMA. It should be clear from the illustration that instead one could similarly alter the BMA to run on a kind of reversed syndrome matrix and to produce error coevaluator Ψ in addition to error locator Λ and error evaluator Ω. It would then perform the same operations as the original EA.

Note that it is not necessary to run the adjusted EA on the reversed syndrome polynomial $\bar{S}$. Giving S as input will result in obtaining the error co-

evaluator Ψ from

$$\Lambda \cdot S = \Omega - \Psi x^{n-k}$$

which can be used just as well in the Forney Algorithm (2.2) to obtain error values.

It is worth mentioning that calculating with the reversed syndrome polynomial and the reciprocal of Λ turns the key equation into the following:

$$\begin{aligned}
\prod_{i=1}^{\epsilon}(x-\alpha^{l_j}) \cdot \bar{S} &= \Lambda^{\mathrm{rec}} \cdot \bar{S} \\
&= x^{\epsilon}\Lambda(x^{-1}) \cdot x^{n-k-1}S(x^{-1}) \\
&= x^{n-k+\epsilon-1}\big(\Lambda \cdot S\big)(x^{-1}) \\
&= x^{n-k+\epsilon-1}\big(\Omega(x^{-1}) - \Psi(x^{-1})x^{-(n-k)}\big) \\
&= x^{n-k}\sum_{j=1}^{\epsilon} e_{l_j}\alpha^{l_j}\prod_{\substack{i=1\\ i\neq j}}^{\epsilon}(x-\alpha^{l_i}) \\
&\quad - \sum_{j=1}^{\epsilon} e_{l_j}\alpha^{(n-k+1)l_j}\prod_{\substack{i=1\\ i\neq j}}^{\epsilon}(x-\alpha^{l_i})
\end{aligned} \tag{2.15}$$

The above suggest an alternate definition of all the polynomials involved

$$\begin{aligned}
\Lambda^* &= \prod_{i=1}^{\epsilon}(x-\alpha^{l_j}), \\
\Omega^* &= \sum_{j=1}^{\epsilon} e_{l_j}\alpha^{l_j}\prod_{\substack{i=1\\ i\neq j}}^{\epsilon}(x-\alpha^{l_i}), \\
\Psi^* &= \sum_{j=1}^{\epsilon} e_{l_j}\alpha^{(n-k+1)l_j}\prod_{\substack{i=1\\ i\neq j}}^{\epsilon}(x-\alpha^{l_i})
\end{aligned}$$

avoiding the awkwardness of α^{-l_i} indicating error locations rather than α^{l_i}. This alternate definition has already been put forth by O'Sullivan (1998) in a slightly different version. He turns the syndrome polynomial into a power series in $1/x$ and can then omit the coevaluator polynomial from the key equation. But dividing the above equation (2.15) as well as $\bar{S}$ by x^{n-k} resulting in $S^* = 1/x \cdot S(x^{-1})$ and running the adjusted EA on 1 and S^* (instead of x^{n-k} and $\bar{S}$) will find Λ^* and Ω^*. O'Sullivan's perspective on the key equation which he also could generalize in (O'Sullivan 1998) might be the better one, and the long confusion whether BMA and EA are the same algorithms or not might be due to the wrong perspective.

2.6 Conclusions

The main contribution of this paper is the clear illustration of all the parallels that can be found between the BMA and the EA in Theorem 2.4. In order to do

this the FIA needed to be extended to calculate the error evaluator and coevaluator as suggested by Theorem 2.3. The coevaluator polynomial does not seem to come in naturally for neither (Berlekamp 1968; Massey 1969) nor (Dornstetter 1987), but because of its inclusion, we can easily go in both directions of how the EA can be adapted to the BMA as well as the other way.

Finally, it can be concluded that the Berlekamp-Massey and the Euclidean algorithm are essentially the same, since they can both be adapted in a simple way to perform the same operations as the other.

Acknowledgement

The authors wish to thank Sørensen and Grønne (1995) for their contributions.

References

Berlekamp, E. R. (1968). *Algebraic Coding Theory.* McGraw-Hill.

Dornstetter, J. L. (1987). On the Equivalence Between Berlekamp's and Euclid's Algorithms. *IEEE Transactions on Information Theory IT-33*, 428–431.

Feng, G.-L. and K. K. Tzeng (1991, September). A Generalization of the Berlekamp-Massey Algorithm for Multisequence Shift-Register Synthesis with Applications to Decoding Cyclic Codes. *IEEE Transactions on Information Theory 37*(5), 1274–1287.

Greuel, G.-M., G. Pfister, and H. Schönemann (1998). Singular Version 1.2 User Manual. In *Reports on Computer Algebra*, Number 21. Centre for Computer Algebra, University of Kaiserslautern.
`http://www.singular.uni-kl.de`.

MacWilliams, F. J. and N. J. A. Sloane (1977). *The Theory of Error-Correcting Codes.* North-Holland Mathematical Library.

Massey, J. L. (1969). Shift-Register synthesis and bch decoding. *IEEE Transactions on Information Theory IT-15*, 122–127.

O'Sullivan, M. E. (1998). The Key Equation for One-Point Codes and Efficient Error Evaluation. manuscript.

Pretzel, O. (1992). *Error-Correcting Codes and Finite Fields.* Oxford Applied Mathematics and Computing Science Series. Oxford University Press.

Sørensen, P. H. and T. Grønne (1995, August). Indkodning og dekodning af cykliske koder i dellegemer. Master's thesis, Technical University of Denmark.

Sugiyama, Y., M. Kasahara, S. Hirasawa, and T. Namekawa (1975). A Method for Solving Key Equation for Decoding Goppa Codes. *Information and Control 27*, 87–99.

3. DECODING REED-MULLER CODES BEYOND HALF THE MINIMUM DISTANCE

Agnes E. Heydtmann*
Department of Mathematics, Building 303, Technical University of Denmark,
DK-2800 Kongens Lyngby, Denmark. Agnes.Heydtmann@mat.dtu.dk

Thomas Jakobsen
IO Interactive APS, Farvergade 2, DK-1463 Copenhagen K, Denmark. tj@ioi.dk

Abstract

Inspired by Sudan's recent algorithm for Reed-Solomon codes we propose an efficient method for decoding r-th order Reed-Muller codes of length 2^m which can correct errors beyond half the minimum distance.

This procedure involves interpolating $Q \in \mathbb{F}_2[x_1, \ldots, x_m, y]$, a polynomial vanishing when evaluated at points in $\mathbb{F}_2^m$ joint with the corresponding received bits. To obtain a list of codewords closest to the received word we need to factor Q considered as an element of the quotient ring of boolean polynomials which is not a unique factorization domain. Therefore we introduce a novel, yet simple polynomial-time factorization algorithm for multivariate boolean polynomials that produces generators for the coset of factors.

Let $p = 2^{-\lambda}$ be the probability of algorithm failure and assume that the weights of a Reed-Muller code are approximately binomially distributed. This assumption is supported by known weight distributions for some short Reed-Muller codes. Then with probability at least $1 - p$, the algorithm corrects

$$\tau \leq \max_{0 \leq \rho \leq m} \min \left\{ 2^m - \sum_{i=0}^{r+\rho} \binom{m}{i} - \lambda, \sum_{i=0}^{\rho} \binom{m}{i} - 1 \right\} \tag{3.1}$$

independently and uniformly distributed errors.

For the $\mathcal{RM}(2,9)$ code for example, the algorithm corrects up to 122 errors with probability at least 0.99 whereas half the minimum distance is 64. Under the above assumption, we can correct up to half the block length asymptotically for fixed r.

*During part of this work Agnes E. Heydtmann was supported by a graduate student scholarship (HSPIII) of the German Academic Exchange Service.

3.1 Introduction

Reed-Muller codes (MacWilliams and Sloane 1977, Chapters 13–15) make a class of codes with a simple construction given by Muller (1954) that allows easy majority logic decoding described by Reed (1954).

At first glance, the structure seems to be very simple and the minimum distance is in fact also easily derived, but beyond this there are still several unanswered questions. The weight distribution and the covering radius, for example, are known only for specific instances of the codes.

Most known algorithms for decoding Reed-Muller codes do not decode beyond half the minimum distance and those that do so, do not have an efficient running time in the general case. Sudan's recent algorithm (Sudan 1997) allows efficient decoding of Reed-Solomon codes (MacWilliams and Sloane 1977, Chapter 10) beyond half their minimum distance. A generalization to some classes of algebraic-geometry codes has been made by Shokrollahi and Wasserman (1999). As there are several similarities in the construction of Reed-Solomon and Reed-Muller codes, it is natural to speculate whether Sudan-like techniques can be applied to the Reed-Muller case.

This paper provides such a generalization. Although the algorithm presented has several similarities to Sudan's algorithm, its requirements and proofs are quite different due to the small size of the underlying field. The algorithm also bears some resemblance to the Welch-Berlekamp algorithm for decoding Reed-Solomon codes (Welch and Berlekamp 1986; Berlekamp 1996). It works best in settings with a low information rate, however the error rate can be correspondingly high. Although when going beyond half the minimum distance, the decoding is not guaranteed to be unique, it is still possible with high probability to correctly determine the codeword sent.

In a sense, decoding of Reed-Muller codes corresponds to the approximation of binary functions by low-degree multivariate polynomials. Consequently, a decoding algorithm which goes beyond half the minimum distance is useful in other applications which are not directly related to error correction. For instance, cryptanalysis of secret-key ciphers (Menezes et al. 1996) often deals with finding approximations, see for example (Jakobsen 1998); here our algorithm could prove to be very useful. In fact, the research resulting in this paper was partly initiated because of an observation by Shimoyama and Kaneko (1998) who realized that a specific quadratic relation of the in- and output of one S-box of the Data Encryption Standard (DES) (Menezes et al. 1996) has the best linear approximation as a factor.

The paper is organized as follows: We proceed in Section 3.2 by defining the Reed-Muller codes and related algebraic concepts. In Section 3.3, the algorithm is presented together with the main results and we give some conjectures about its running time and correctness for random error patterns. The conjectures are related to the generally unknown weight distribution of Reed-Muller codes. They are supported by an implementation of the algorithm and known weight-distributions for short-length Reed-Muller codes. In Section 3.4 we proceed to prove the correctness of the algorithm. Section 3.5 deals with the error-correcting capabilities and Section 3.6 covers complexity issues. We conclude in Section 3.7.

3.2 Preliminaries

We begin by defining Reed-Muller codes in terms of elements of a quotient ring.

Definition 3.1 *1. The quotient ring*

$$R_{[x_1,\ldots,x_m]} = \mathbb{F}_2[x_1,\ldots,x_m] \,/\, \langle x_1^2 + x_1, \ldots, x_m^2 + x_m \rangle$$

is the ring of boolean polynomials *in m variables $x_1,\ldots,x_m$. Elements of the ring are called* boolean polynomials *and in our notation we naturally identify cosets and polynomials in $\mathbb{F}_2[x_1,\ldots,x_m]$ that are linear in each of the m variables. Furthermore, monomials in $\mathbb{F}_2[x_1,\ldots,x_m]$ that are linear in each of the m variables and their cosets in $R_{[x_1,\ldots,x_m]}$ are called* boolean monomials.

2. *The* degree *of a coset $f \in R_{[x_1,\ldots,x_m]}$ denoted by $\deg f$, is the degree of the unique element that is linear in each of the m variables.*

3. *The* r-th order Reed-Muller code *of length 2^m with $0 \le r \le m$, is the set*

$$\mathcal{RM}(r,m) = \left\{ (f(P_1),\ldots,f(P_{2^m})) \mid f \in R_{[x_1,\ldots,x_m]}, \deg f \le r \right\}$$

where $P_1,\ldots,P_{2^m}$ are the distinct elements of $\mathbb{F}_2^m$. We associate naturally $f \in R_{[x_1,\ldots,x_m]}$, $\deg f \le r$ with $\mathcal{RM}(r,m)$ words.

Note that each coset contains exactly one polynomial which is linear in each of the m variables and that all elements of one coset in $R_{[x_1,\ldots,x_m]}$ are equal when considered as functions $\mathbb{F}_2^m \to \mathbb{F}_2$. Therefore $\mathcal{RM}(r,m)$ codes are well defined.

The boolean monomials in $R_{[x_1,\ldots,x_m]}$ evaluate to linearly independent vectors in $\mathbb{F}_2^{2^m}$ and those of degree r result in codewords of minimal weight. Thus the $\mathcal{RM}(r,m)$ code is a linear $[2^m, \binom{m}{0} + \cdots + \binom{m}{r}, 2^{m-r}]$ code. We will also denote the dimension of $\mathcal{RM}(r,m)$ by $k_r = \sum_{i=0}^{r} \binom{m}{i}$.

The ring of boolean polynomials $R_{[x_1,\ldots,x_m]}$ is not even an integral domain as $h \cdot (h+1) = 0$ for any $h \in R_{[x_1,\ldots,x_m]}$. Therefore the ring is certainly not a unique factorization domain. Nevertheless we define the following:

Definition 3.2 *Let $R_{[x_1,\ldots,x_m]}$ be a ring of boolean polynomials and let $h, f \in R_{[x_1,\ldots,x_m]}$. The boolean polynomial f is said to be a* factor *of h or to* divide *h, if there exists $g \in R_{[x_1,\ldots,x_m]}$ such that*

$$h = f \cdot g \ .$$

Certain factors of the $Q \in R_{[x_1,\ldots,x_m,y]}$ which we will interpolate in Algorithm 3.1 will correspond to $\mathcal{RM}(r,m)$ codewords closest to the received word. Therefore it is convenient that the following theorem easily characterizes all factors of any boolean polynomial.

Theorem 3.1 *A boolean polynomial $f \in R_{[x_1,\ldots,x_m]}$ divides a boolean polynomial h if and only if*

$$(f+1) \cdot h = 0 \ .$$

Proof : $\Rightarrow$: Assume that there exists a boolean polynomial g such that $h = f \cdot g$. Then

$$(f+1) \cdot h = (f+1) \cdot f \cdot g = (f^2 + f) \cdot g = 0$$

where the last equality holds since $(f^2 + f) \in \langle x_1^2 + x_1, \ldots, x_m^2 + x_m \rangle$.
$\Leftarrow$: Assume that $(f+1) \cdot h = 0$. Then $h = fh$, i.e. there exists a boolean polynomial $g = h$ such that $h = fg$ which means that f is a factor of h by the definition. □

Theorem 3.1 implies that the factors of a boolean polynomial h form the coset $1 + \{f \in R_{[x_1,\ldots,x_m]} | f \cdot h = 0\}$ of the ideal $\{f \in R_{[x_1,\ldots,x_m]} | f \cdot h = 0\}$ in $R_{[x_1,\ldots,x_m]}$. A factorization algorithm for multivariate, boolean polynomials is immediately obtained: Simply solve the corresponding homogeneous linear system of equations in the coefficients of the factor. In this way it is even straight forward to restrict the solutions to be bounded in their degree.

Let τ denote the maximum number of errors that should be corrected by the algorithm and let ρ be the smallest integer for which

$$\tau < k_\rho \ .$$

In order to interpolate the boolean polynomial $Q \in R_{[x_1,\ldots,x_m,y]}$ from Algorithm 3.1 that should help us to decode up to τ errors, we need the notion of the order of a boolean polynomial of degree at most ρ in our original ring $R_{[x_1,\ldots,x_m]}$ with respect to an arbitrary basis of this subvector space.

Definition 3.3 *Let $p_0, \ldots, p_{k_\rho - 1} \in R_{[x_1,\ldots,x_m]}$ denote k_ρ linearly independent, boolean polynomials with $\deg p_j \leq \rho$. Let further $0 \neq f \in R_{[x_1,\ldots,x_m]}$ of degree at most ρ. Then f can be expressed uniquely as*

$$f = \sum_{j=0}^{s} c_j p_j$$

where $c_j \in \mathbb{F}_2$, $s \leq k_\rho - 1$ and $c_s \neq 0$. Define by the order *of f*

$$\operatorname{ord} f = s \ .$$

Note that the above does not define an ordering on $R_{[x_1,\ldots,x_m]}$ since the relation is not antisymmetric. Clearly, $\operatorname{ord} p_j = j$ and if p_j is a boolean monomial for all j, then the order of a boolean polynomial is simply the maximum order of its boolean monomials.

For Algorithm 3.1 to run correctly the error pattern has to be random in a certain sense. Therefore we introduce the following concepts and notation:

Definition 3.4 *Let $\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{F}_2^n$ be an n-bit word. The* weight *of* $\mathbf{a}$ *denoted by* $\mathrm{w}(\mathbf{a})$ *is the number of non-zero positions in* $\mathbf{a}$. *The* support *of* $\mathbf{a}$ *is the set of its non-zero positions*

$$\operatorname{supp}(\mathbf{a}) = \{i | \, a_i \neq 0\} \ .$$

Clearly, $\mathrm{w}(\mathbf{a}) = |\operatorname{supp}(\mathbf{a})|$. *We say that a word* $\mathbf{a} \in \mathbb{F}_2^n$ covers *another word* $\mathbf{b} \in \mathbb{F}_2^n$ *or that* $\mathbf{b}$ *is* covered *by* $\mathbf{a}$ *when* $\operatorname{supp}(\mathbf{b}) \subseteq \operatorname{supp}(\mathbf{a})$.

3.3 The Decoding Algorithm

In this section we present the algorithm as well as the main results.

Suppose $\mathbf{y} = (y_1, y_2, \ldots, y_{2^m})$ corresponding to a $\mathcal{RM}(r,m)$ word with at most $\tau = k_\rho - 1$ errors is received. Under certain conditions, the following algorithm decodes this received word correctly.

Input: $\mathbf{y}$, r, ρ, and m
Output: all $f \in R_{[x_1,\ldots,x_m]}$ corresponding to $\mathcal{RM}(r,m)$ codewords closest to $\mathbf{y}$

1: Choose $p_0, \ldots, p_{k_\rho - 1} \in R_{[x_1,\ldots,x_m]}$ linearly independent with $\deg p_j \leq \rho$.
2: Find non-zero $Q = yQ_1 + Q_2 \in R_{[x_1,\ldots,x_m,y]}$, $Q_1, Q_2 \in R_{[x_1,\ldots,x_m]}$ with $\operatorname{ord} Q_1$ lowest possible and $\deg Q_2 \leq r + \rho$ such that $Q(P_i, y_i) = 0 \; \forall i$.
3: Find $f \in R_{[x_1,\ldots,x_m]}$ with $\deg f \leq r$ such that $f(P_i) = y_i, \forall i : Q_1(P_i) = 1$.
4: Return those f that correspond to codewords closest to $\mathbf{y}$.

Algorithm 3.1: Reed-Muller Decoder

The conditions for correctness of Algorithm 3.1 is given by Theorem 3.2. It will be proven in Section 3.4.

Theorem 3.2 (Correctness) *Suppose that $f_{\mathbf{c}} \in R_{[x_1,\ldots,x_m]}$ corresponds to a $\mathcal{RM}(r,m)$ codeword $\mathbf{c}$ closest to $\mathbf{y}$. If the error pattern $\mathbf{e} = \mathbf{c} + \mathbf{y}$ does not cover a non-zero $\mathcal{RM}(r+\rho, m)$ codeword and $\mathrm{w}(\mathbf{e}) \leq \tau < k_\rho$, then Algorithm 3.1 returns $f_{\mathbf{c}}$.*

Of course we have to ask how often the condition that the error pattern $\mathbf{e} = \mathbf{c} + \mathbf{y}$ does not cover a non-zero $\mathcal{RM}(r+\rho, m)$ codeword will be satisfied when errors occur independently and are uniformly distributed. Certainly for τ less than the code's minimum distance $2^{m-r-\rho}$, the error pattern can never cover a nonzero $\mathcal{RM}(r+\rho, m)$ codeword, but then we are not beyond half the minimum distance either. Conjecture 3.1 addresses the problem for larger τ and will be discussed further in Section 3.5.

Conjecture 3.1 *Let $r + \rho \gg 0$ and $\tau < n - k_{r+\rho}$. Denote the probability that a random error pattern of at most τ errors covers a non-zero $\mathcal{RM}(r+\rho, m)$ codeword by π_1. Then*

$$\lim_{m\to\infty} \left| \pi_1 - 2^{k_{r+\rho} - 2^m + \tau} \right| = 0 \ .$$

We shall see in Section 3.5 that if it can be proven that for $r + \rho \gg 0$ the weights of $\mathcal{RM}(r+\rho, m)$ are close to being binomially distributed, then we can use $2^{k_{r+\rho} - 2^m + \tau}$ as an approximative bound to π_1. We include the condition $r + \rho \gg 0$, since obviously the weights of $\mathcal{RM}(0,m)$ and $\mathcal{RM}(1,m)$ for example, are far from being binomially distributed.

Theorem 3.3 gives partial insight to the efficiency of the algorithm.

Theorem 3.3 (Efficiency) *Suppose that $f_{\mathbf{c}} \in R_{[x_1,\ldots,x_m]}$ corresponds to a codeword $\mathbf{c}$ of $\mathcal{RM}(r,m)$ closest to $\mathbf{y}$ and that the error pattern $\mathbf{e} = \mathbf{c} + \mathbf{y}$ does not*

cover a non-zero $\mathcal{RM}(r+\rho, m)$ codeword. In addition, let π_2 denote the probability that the $\mathcal{RM}(\rho, m)$ codeword corresponding to the boolean polynomial Q_1 obtained in Step 2 of Algorithm 3.1 is covered by some $\mathcal{RM}(r, m)$ codeword $\neq (1, 1, \ldots, 1)$. Then with probability at least $1-\pi_2$, the algorithm has a running time of $O(n^3)$.

The above theorem is proven in Section 3.6. Coupled with Conjecture 3.2 which is further discussed in the same section, we have an efficient algorithm for decoding Reed-Muller codes beyond half the minimum distance.

Conjecture 3.2 *Let $I \subseteq \{0, 1, \ldots, 2^m - 1\}$ be the set of error positions corresponding to the $\mathcal{RM}(r, m)$ codewords $\mathbf{c}$ closest to $\mathbf{y}$. Further we assume that every error pattern $\mathbf{e} = \mathbf{c} + \mathbf{y}$ has weight $< k_\rho$ and does not cover a non-zero $\mathcal{RM}(r+\rho, m)$ codeword. Let $\mathbf{q}_1$ denote a randomly chosen $\mathcal{RM}(\rho, m)$ codeword such that its components $q_{1i} = 0$ for $i \in I$. Let π_2 denote the probability that $\mathbf{q}_1$ is covered by some $\mathcal{RM}(r, m)$ codeword $\neq (1, 1, \ldots, 1)$. If $\rho > r$, then $\pi_2 \to 0$ for $m \to \infty$.*

Section 3.6 should make it clear that the probabilities π_2 mentioned in Theorem 3.3 and Conjecture 3.2 are indeed the same. Also, we show that π_2 is the same as the probability that the random boolean polynomial $\mathbf{q}_1$ has factors of low degree. It appears that this probability is negligible.

The following is an example of a Reed-Muller code capable of being decoded beyond half the minimum distance.

Example 3.1 Consider $\mathcal{RM}(1, 6)$ which has half minimum distance 16 and let $\rho = 2$ which enables us to correct up to 21 errors if there is no non-zero $\mathcal{RM}(3, 6)$ codeword covered by the error pattern. Suppose we send the codeword $\mathbf{c}$ corresponding to $f = x_1+x_2+x_4$ and $\mathbf{y} = \mathbf{c}+\mathbf{e}$ is received corresponding to the occurrence of 20 errors as indicated by $\mathbf{e}$:

$$\begin{array}{cl} & \mathbf{c} : 01100110100110010110011010011001011001101001100101100110100110001 \\ + & \mathbf{e} : 10000000010100011001000011001010000010100010110000110001000010001 \\ \hline = & \mathbf{y} : 11100110110010001111011001010011011011001011010101010111100100000 \end{array}$$

Choosing the p_j's as boolean monomials of increasing order, Step 2 of the algorithm interpolates

$$\begin{aligned} Q = y(&x_2x_3+x_1x_4+x_3x_4+x_1x_5+x_2x_5+x_3x_5+x_4x_5+x_1x_6+x_2x_6+x_4) \\ &+x_1x_2x_3+x_1x_2x_4+x_1x_3x_4+x_1x_3x_5+x_2x_3x_5+x_3x_4x_5+x_1x_4x_6 \\ &+x_2x_4x_6+x_2x_3+x_1x_4+x_2x_4+x_3x_4+x_1x_5+x_2x_5+x_4x_5+x_1x_6 \\ &+x_2x_6+x_4 \end{aligned}$$

which has the following unique factorization of the form $(y+f)Q_1$ with $\deg f \leq r$:

$$\begin{aligned} Q = &(y+x_1+x_2+x_4) \\ &(x_2x_3+x_1x_4+x_3x_4+x_1x_5+x_2x_5+x_3x_5+x_4x_5+x_1x_6+x_2x_6+x_4) \end{aligned}$$

The first factor corresponds to the sent codeword $\mathbf{y}$.

The code $\mathcal{RM}(1,6)$ was chosen to be the smallest example where more errors than half the minimum distance can be corrected with Algorithm 3.1 in practice. Generally, one should not attempt to decode such a small code much beyond half the minimum distance as only few error patterns satisfy the conditions of Theorem 3.2. In Section 3.7 we discuss the decoding of the $\mathcal{RM}(2,9)$ code which is more robust due to its larger size.

3.4 Correctness of the Algorithm

Let boolean polynomials $p_0, \ldots, p_{k_\rho - 1}$ be given as in Definition 3.3 and the order of elements of $R_{[x_1,\ldots,x_m]}$ of degree at most ρ is thereby defined. Let further $f_{\mathbf{c}} \in R_{[x_1,\ldots,x_m]}$ correspond to an $\mathcal{RM}(r,m)$ codeword $\mathbf{c}$ at distance $t \leq \tau < k_\rho$ to the received word $\mathbf{y} = (y_1, \ldots, y_{2^m})$ such that there exists no $\mathcal{RM}(r,m)$ codeword closer to $\mathbf{y}$.

We will investigate two types of boolean polynomials $Q, \tilde{Q} \in R_{[x_1,\ldots,x_m,y]}$ interpolating the points (P_i, y_i). Therefore we introduce the following notation: By $Q = yQ_1 + Q_2$, $Q_1, Q_2 \in R_{[x_1,\ldots,x_m]}$ we denote a non-zero boolean polynomial that vanishes at (P_i, y_i) with $\operatorname{ord} Q_1$ smallest possible and $\deg Q_2 \leq r + \rho$. By $\tilde{Q} = (y + f_{\mathbf{c}})\tilde{Q}_1$, $\tilde{Q}_1 \in R_{[x_1,\ldots,x_m]}$ we denote another non-zero boolean polynomial that vanishes at (P_i, y_i) with $\operatorname{ord} \tilde{Q}_1$ smallest possible. Note that $\deg f_{\mathbf{c}}\tilde{Q}_1 \leq r + \rho$ and $\tilde{Q}_1(P_i) = 0$ whenever $f_{\mathbf{c}}(P_i) \neq y_i$. This means that the existence of $\tilde{Q}$ implies the existence of Q and that the boolean polynomial $\tilde{Q}_1$ may be thought of as an error locator.

Both Q and $\tilde{Q}$ are uniquely determined: Let $Q = yQ_1 + Q_2, Q' = yQ_1' + Q_2' \in R_{[x_1,\ldots,x_m,y]}$, both vanishing when evaluated in (P_i, y_i) and such that $\operatorname{ord} Q_1 = \operatorname{ord} Q_1'$ lowest possible, $\deg Q_2, \deg Q_2' \leq r + \rho$. But

$$Q - Q' = y(Q_1 - Q_1') + (Q_2 - Q_2')$$

is a new boolean polynomial vanishing when evaluated in (P_i, y_i). Either $\operatorname{ord} Q_1 - Q_1' < \operatorname{ord} Q_1 = \operatorname{ord} Q_1'$ and $\deg Q_2 - Q_2' \leq r + \rho$ which results in a contradiction to the definition of Q, or $Q_1 = Q_1'$. In that case we have to conclude that the boolean polynomial $Q_2 - Q_2'$ as an element in $R_{[x_1,\ldots,x_m,y]}$ vanishes at all (P_i, y_i). But that means that as an element of $R_{[x_1,\ldots,x_m]}$ it vanishes at all points in $\mathbb{F}_2^m$ and must therefore equal the 0-polynomial. That implies $Q_2 = Q_2'$ and we must have had $Q = Q'$ in the first place. Similarly one can prove the uniqueness of $\tilde{Q}$. We will proceed to show that, under some conditions, Q and $\tilde{Q}$ are in fact identical.

Theorem 3.4 *If the error pattern $\mathbf{e} = \mathbf{c} + \mathbf{y}$ does not cover a $\mathcal{RM}(m, r+\rho)$ word, then $Q = \tilde{Q}$ and $\operatorname{ord} Q_1 = \operatorname{ord} \tilde{Q}_1 \leq t$, where $t \leq \tau < k_\rho$ is the number of errors.*

In the proof of the above we will make use of Lemma 3.1. We start by introducing some more notation. Let $l_1, \ldots, l_t$ denote the error positions and $I_j = \{l_{j+1}, \ldots, l_t\}$ for $0 \leq j \leq t$. Denote by $Q^{(j)} = yQ_1^{(j)} + Q_2^{(j)}$, $Q_1^{(j)}, Q_2^{(j)} \in R_{[x_1,\ldots,x_m]}$ the boolean polynomial that vanishes at (P_i, y_i) for $i \notin I_j$ (i.e., at the

first j error positions and at every non-erroneous position) with $\operatorname{ord} Q_1^{(j)}$ smallest possible and $\deg Q_2^{(j)} \leq r+\rho$. Analogously, denote by $\tilde{Q}^{(j)} = (y+f_{\mathbf{c}})\tilde{Q}_1^{(j)}$, $\tilde{Q}_1^{(j)} \in R_{[x_1,\ldots,x_m]}$ the boolean polynomial that vanishes at (P_i, y_i) for $i \notin I_j$ with $\operatorname{ord} \tilde{Q}_1^{(j)}$ smallest possible. Note that $\deg f_{\mathbf{c}}\tilde{Q}_1^{(j)} \leq r+\rho$ and $\tilde{Q}_1^{(j)}(P_i) = 0$ for $l_1, \ldots, l_j$. This means that the existence of $\tilde{Q}^{(j)}$ implies the existence of $Q^{(j)}$ and that the boolean polynomial $\tilde{Q}_1^{(j)}$ is an error locator for the first j error positions.

Under the conditions of Theorem 3.4 both $Q^{(j)}$ and $\tilde{Q}^{(j)}$ are again uniquely determined. Let $Q^{(j)} = yQ_1^{(j)} + Q_2^{(j)}, Q'^{(j)} = yQ'^{(j)}_1 + Q'^{(j)}_2 \in R_{[x_1,\ldots,x_m,y]}$, both vanishing when evaluated in (P_i, y_i) for $i \notin I_j$ and such that $\operatorname{ord} Q_1^{(j)} = \operatorname{ord} Q'^{(j)}_1$ lowest possible, $\deg Q_2^{(j)}, \deg Q'^{(j)}_2 \leq r+\rho$. But

$$Q^{(j)} - Q'^{(j)} = y(Q_1^{(j)} - Q'^{(j)}_1) + (Q_2^{(j)} - Q'^{(j)}_2)$$

is a new boolean polynomial vanishing when evaluated in (P_i, y_i) for $i \notin I_j$. Either $\operatorname{ord} Q_1^{(j)} - Q'^{(j)}_1 < \operatorname{ord} Q_1^{(j)} = \operatorname{ord} Q'^{(j)}_1$ and $\deg Q_2^{(j)} - Q'^{(j)}_2 \leq r+\rho$ which results in a contradiction to the definition of $Q^{(j)}$, or $Q_1^{(j)} = Q'^{(j)}_1$. In that case we have to conclude that the boolean polynomial $Q_2^{(j)} - Q'^{(j)}_2$ as element in $R_{[x_1,\ldots,x_m,y]}$ vanishes at (P_i, y_i) for $i \notin I_j$. But that means that the $\mathcal{RM}(r+\rho, m)$ codeword corresponding to $Q_2^{(j)} - Q'^{(j)}_2$ is covered by $\mathbf{e}$. We must have had $Q^{(j)} = Q'^{(j)}$ in the first place. Similarly one can prove the uniqueness of $\tilde{Q}^{(j)}$.

Lemma 3.1 *Let* $Q^{(j)} = \tilde{Q}^{(j)}$ *with* $\operatorname{ord} Q_1^{(j)} = \operatorname{ord} \tilde{Q}_1^{(j)} = j$.

1. *If* $Q^{(j)}(P_i, y_i) = 0$ *for all* $i \in I_j$, *then* $Q = Q^{(j)} = \tilde{Q}^{(j)} = \tilde{Q}$.
2. *If* $Q^{(j)}(P_i, y_i) \neq 0$ *for some* $i \in I_j$ *(without loss of generality* $i = l_{j+1}$*), then* $Q^{(j+1)} = \tilde{Q}^{(j+1)}$ *with* $\operatorname{ord} Q_1^{(j+1)} = \operatorname{ord} \tilde{Q}_1^{(j+1)} = j+1$.

Proof : Consider the two cases.

1. If $Q^{(j)}(P_i, y_i) = 0$ for all $i \in I_j$, then $Q^{(j)} = Q$ and by assumption also $Q^{(j)} = \tilde{Q}^{(j)}$ implying $\tilde{Q}^{(j)} = \tilde{Q}$. All in all, we get $Q = Q^{(j)} = \tilde{Q}^{(j)} = \tilde{Q}$.
2. If $Q^{(j)}(P_i, y_i) \neq 0$ for an $i \in I_j$, then without loss of generality we may assume that $i = l_{j+1}$.

   Consider the boolean polynomials $Q^{(j+1)}$ and $\tilde{Q}^{(j+1)}$ which vanish at (P_i, y_i) for $i \notin I_j$. Note that $\tilde{Q}_1^{(j+1)}$ has to be chosen such that it evaluates to 0 at P_i for $i = l_1, \ldots, l_{j+1}$, but this is possible as a linear combination of the $j+2$ boolean polynomials $p_0, \ldots, p_{j+1}$. Therefore $\operatorname{ord} \tilde{Q}_1^{(j+1)} \leq j+1$. Further, $\operatorname{ord} \tilde{Q}_1^{(j+1)} < j+1$ would contradict the uniqueness of $\tilde{Q}^{(j)}$. Therefore $\operatorname{ord} \tilde{Q}_1^{(j+1)} = j+1$.

   On the other hand the existence of $Q^{(j+1)}$ is assured by the existence of $\tilde{Q}^{(j+1)}$. The inequality $\operatorname{ord} Q_1^{(j+1)} < j+1$ would contradict the uniqueness of $Q^{(j)}$. Also $\operatorname{ord} Q_1^{(j+1)} > j+1$ is impossible as in that case $\tilde{Q}^{(j+1)}$ would be of the same type as $Q^{(j+1)}$, but with $\operatorname{ord} \tilde{Q}_1^{(j+1)} < \operatorname{ord} Q^{(j+1)}$. I.e. $\operatorname{ord} Q_1^{(j+1)} = j+1 = \operatorname{ord} \tilde{Q}_1^{(j+1)}$.

   Consider now the boolean polynomial $Q' = Q^{(j+1)} - \tilde{Q}^{(j+1)} = yQ'_1 + Q'_2$ with $Q'_1, Q'_2 \in R_{[x_1,\ldots,x_m]}$. Since both $Q_1^{(j+1)}$ and $\tilde{Q}_1^{(j+1)}$ have order $j+1$,

$Q_1^{(j+1)} - \tilde{Q}_1^{(j+1)}$ is already a linear combination of $p_0, \ldots, p_j$ and $\operatorname{ord} Q_1' < j+1$. In addition, we have $Q' \neq Q^{(j)}$ as they evaluate to 0 and 1 at $(P_{l_{j+1}}, y_{l_{j+1}})$ respectively, but they both vanish for (P_i, y_i), $i \notin I_j$. This contradicts again the uniqueness of $Q^{(j)}$. Therefore we must have the equality $Q^{(j+1)} = \tilde{Q}^{(j+1)}$ with $\operatorname{ord} Q_1^{(j+1)} = \operatorname{ord} \tilde{Q}_1^{(j+1)} = j+1$.

□

Proof of Theorem 3.4: We prove the theorem by induction on the number $j = 0, \ldots, t$ of error positions that are interpolated into $Q^{(j)}$ in addition to the non-erroneous positions.

$j = 0$**:** Obviously we can choose $\tilde{Q}^{(0)} = (y + f_{\mathbf{c}}) \cdot p_0$ and $Q^{(0)}$ must be of the form $Q^{(0)} = p_0 y + h$ with $h \in R_{[x_1,\ldots,x_m]}$, $\deg h \leq r + \rho$. Both boolean polynomials are of order 0. Now $\tilde{Q}^{(0)} - Q^{(0)} = p_0 \cdot f_{\mathbf{c}} - h$ is a new boolean polynomial vanishing at $i \notin I_0$. Then $(f_b c \cdot p_0 + h)(P_i) = 0$ when i is is a non-erroneous position, but $f_{\mathbf{c}} \cdot p_0 + h \neq 0$ would imply that the support of the non-zero $\mathcal{RM}(m, r+\rho)$ word corresponding to $f_{\mathbf{c}} \cdot p_0 + h$ is contained in the support of the error pattern $\mathbf{e}$, i.e. that $\mathbf{e}$ is covered by that word. This contradicts our assumptions and therefore $Q^{(0)} = \tilde{Q}^{(0)}$.

$j \to j+1$**:** Suppose we have $Q^{(j)} = \tilde{Q}^{(j)}$ with $\operatorname{ord} Q_1^{(j)} = \operatorname{ord} \tilde{Q}_1^{(j)} = j$. By Lemma 3.1 this implies that either $Q = Q^{(j)} = \tilde{Q}^{(j)} = \tilde{Q}$ or after renaming the remaining error positions such that $Q^{(j)}(P_{l_{j+1}}, y_{l_{j+1}}) = 1$, we have $Q^{(j+1)} = \tilde{Q}^{(j+1)}$ with $\operatorname{ord} Q_1^{(j+1)} = \operatorname{ord} \tilde{Q}_1^{(j+1)} = j+1$.

After having interpolated up to (P_{l_t}, y_{l_t}), we are done and obtain $Q = \tilde{Q}$ with $\operatorname{ord} Q_1 \leq t$. Note that the order in which we interpolate error positions has no relevance. □

We can now prove that Algorithm 3.1 works in the sense of Theorem 3.2.

Proof of Theorem 3.2: The conditions of Theorem 3.2 and Theorem 3.4 are the same. According to the latter, we have

$$Q = yQ_1 + Q_2 = (y + f_{\mathbf{c}})Q_1$$

after Step 2 in Algorithm 3.1. As mentioned earlier, the boolean polynomial Q_1 may be thought of as an error locator since $Q_1(P_i) = 1$ implies that position i is error-free. Consequently, $f_{\mathbf{c}}$ can be obtained when constructing $f \in R_{[x_1,\ldots,x_m]}$ with $\deg f \leq r$ such that $f(P_i) = y_i$ for those i where $Q_1(P_i) = 1$. In other words, $f_{\mathbf{c}}$ is a solution to Step 3 and by its definition also to Step 4. □

Note that Theorem 3.2 implies that if $\mathbf{c}$ and $\mathbf{c}'$ are distinct $\mathcal{RM}(r, m)$ codewords closest to $\mathbf{y}$, then

$$Q = (y + f_{\mathbf{c}})Q_1 = (y + f_{\mathbf{c}'})Q_1 \ ,$$

and both $f_{\mathbf{c}}$ and $f_{\mathbf{c}'}$ are solutions to Step 4.

Further, Step 3 in Algorithm 3.1 could be formulated in two different ways: One can apply Theorem 3.1 and find factors of the form $y + f$ of Q by solving $(y + f + 1)Q = 0$. This is similar to the approach taken in Sudan's algorithm (Sudan 1997). On the other hand $Q = yQ_1 + Q_2 = (y + f_{\mathbf{c}})Q_1$ implies $Q_2 = f_{\mathbf{c}}Q_1$ and a simpler system of linear equations can be obtained. This is similar to the equation solved by the Welch-Berlekamp algorithm (Welch and Berlekamp 1986).

3.5 Error Correction Capability

In this section we will discuss the probability that a random error pattern $\mathbf{e} = \mathbf{c} + \mathbf{y}$ with weight at most $\tau < k_\rho$ will cover a non-zero $\mathcal{RM}(r + \rho, m)$ word. It is essential for the error correction capability of our algorithm that this probability is small.

In the following let $A_0, \ldots, A_n$ be the weight distribution of an arbitrary binary linear code C with parameters $[n, k, d]$. From (MacWilliams and Sloane 1977, Chapter 9,§10) we know that a good rule of thumb is

$$A_w \approx 2^{k-n}\binom{n}{w} \quad . \tag{3.2}$$

This rather vague statement is made more precise by Sidelnikov's theorem which shows that the cumulative distribution function associated with the code C

$$A(x) = \frac{1}{2^k}\sum_{0 \le w \le x}^{n} A_w$$

is in some sense close to the normal cumulative distribution function. But this means that in this sense the function A is also close to the function

$$B(x) = \frac{1}{2^n}\sum_{0 \le w \le x}^{n} \binom{n}{w}$$

which is the cumulative distribution function of the trivial $[n, n, 1]$ code. This justifies (3.2) somewhat.

The weight distribution of $\mathcal{RM}(5, 9)$ is known due to Sugita et al. (1996) and we compare its cumulative distribution function to B in Figure 3.1 (a). The closeness of their course supports the view that Reed-Muller codes are like random binary codes and random binary linear codes among others close to what we will loosely call *binomially distributed*.

For such a close to binomially distributed $[n, k, d]$ code C we can estimate the probability π_1 that a random error pattern of weight at most τ covers a non-zero codeword:

$$\begin{aligned}\pi_1 &= \frac{|\{\mathbf{e} \in \mathbb{F}_2^n | \mathrm{w}(\mathbf{e}) \le \tau, \exists\, \mathbf{c} \in C, \mathbf{c} \ne 0 : \mathrm{supp}\,(\mathbf{c}) \subseteq \mathrm{supp}\,(\mathbf{e})\}|}{|\{\mathbf{e} \in \mathbb{F}_2^n | \mathrm{w}(\mathbf{e}) \le \tau\}|} \\ &\le \left(\sum_{t=1}^{\tau}\sum_{w=d}^{t} A_w \binom{n-w}{t-w}\right) \cdot \left(\sum_{t=0}^{\tau}\binom{n}{t}\right)^{-1}\end{aligned} \tag{3.3}$$

$$
\begin{aligned}
&\overset{(3.2)}{\approx} \left(\sum_{t=1}^{\tau}\sum_{w=d}^{t} 2^{k-n}\binom{n}{w}\binom{n-w}{t-w}\right)\cdot\left(\sum_{t=0}^{\tau}\binom{n}{t}\right)^{-1}\\
&\le 2^{k-n}\left(\sum_{t=0}^{\tau}\sum_{w=0}^{t}\frac{n!}{w!(n-w)!}\frac{(n-w)!}{(t-w)!(n-t)!}\right)\cdot\left(\sum_{t=0}^{\tau}\binom{n}{t}\right)^{-1}\\
&= 2^{k-n}\left(\sum_{t=0}^{\tau}\frac{n!}{t!(n-t)!}\sum_{w=0}^{t}\frac{t!}{w!(t-w)!}\right)\cdot\left(\sum_{t=0}^{\tau}\binom{n}{t}\right)^{-1}\\
&= 2^{k-n}\left(\sum_{t=0}^{\tau}\binom{n}{t}\sum_{w=0}^{t}\binom{t}{w}\right)\cdot\left(\sum_{t=0}^{\tau}\binom{n}{t}\right)^{-1}\\
&= 2^{k-n}\left(\sum_{t=0}^{\tau}\binom{n}{t}2^{t}\right)\cdot\left(\sum_{t=0}^{\tau}\binom{n}{t}\right)^{-1}\\
&\le 2^{k-n+\tau}\quad . \qquad (3.4)
\end{aligned}
$$

The inequality in (3.3) is given by counting error patterns with the (un-)desired property by choosing codewords and complementing the support by adding additional non-zero bits.

In Figure 3.1 (b) we compare the accurate upper bound (3.3) for π_1 in the case of the $\mathcal{RM}(5,9)$ code that we can calculate due to the known weights A_w and the approximate bound (3.4). This plot demonstrates that the two bounds are close to each other up to $\tau = 130 = 2^9 - k_5$ in an absolute sense, but not relatively. For $\tau > 130$ both bounds rise quickly to large numbers. The reason for this is that the estimate in (3.3) turns out to be extremely rough for big τ, as different codewords from the considered code can be complemented to give the same error pattern of some weight.

Note that for $r = 0, 1, 2$ the entire weight distributions of $\mathcal{RM}(r,m)$ and their duals are known (MacWilliams and Sloane 1977, Chapter 15). For general $\mathcal{RM}(r,m)$ the numbers $A_0, \ldots, A_{5\cdot 2^{m-r-1}}$ are known (Kasami et al. 1976).

Consequently, if we can prove that Reed-Muller codes of a certain size are approximately binomially distributed which Sidelnikov's theorem is unfortunately too weak for, then the two bounds (3.3) and (3.4) are close to each other and the latter can be used as an approximate bound on π_1 as claimed in Conjecture 3.1.

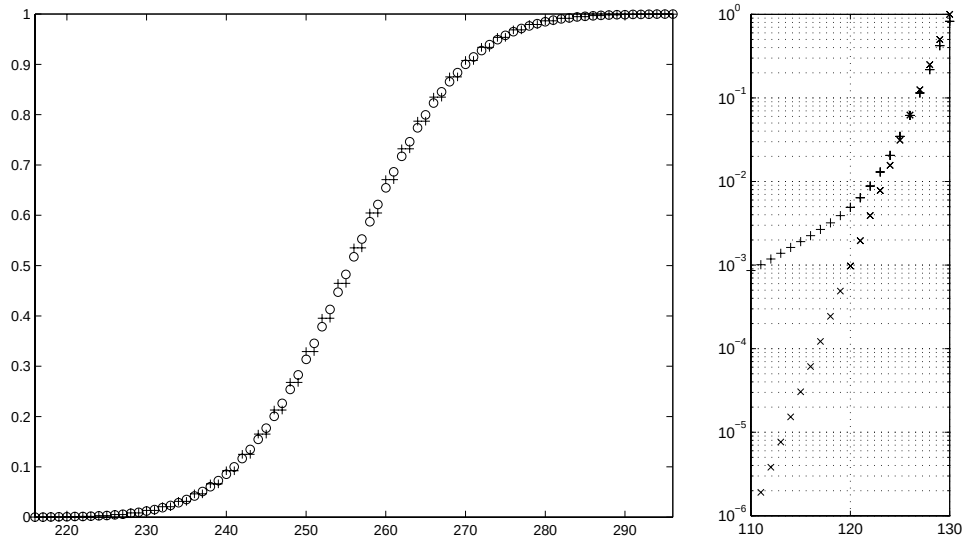
We can now state the following sufficient conditions that the algorithm decodes correctly.

Proposition 3.1 *Consider the Reed-Muller code $\mathcal{RM}(r,m)$ and let*

$$\tau = \max_{0\le\rho\le m} \min\left\{2^m - k_{r+\rho} - \lambda, k_\rho - 1\right\}, \quad \lambda \in \mathbb{N}_0 \quad . \qquad (3.5)$$

If $\pi_1 \le 2^{k_{r+\rho}-2^m+\tau}$, then Algorithm 3.1 correctly decodes up to τ random errors that are independently and uniformly distributed, with probability of failure at most $2^{-\lambda}$.

Proof: Obviously if $\pi_1 \le 2^{k_{r+\rho}-2^m+\tau}$ and $\tau \le 2^m - k_{r+\rho} - \lambda$, then $\pi_1 \le 2^{-\lambda}$. But $\tau < k_\rho$ is also required by Theorem 3.2. □



(a) (b)

Figure 3.1: (a) The cumulative distribution function A '+' for the code $\mathcal{RM}(5,9)$ and B 'o' for $x \in \{216,\ldots,296\}$ (b) The bound (3.3) '+' for the code $\mathcal{RM}(5,9)$ and the approximate bound (3.4) '×' for $\tau \in \{110,\ldots,130\}$

From (3.5) we immediately get (3.1). Check Table 3.1 for a comparison of half the minimum distance 2^{m-r-1} and the numbers τ. For small m or high r, the number τ is lower than half the minimum distance, because the probability that the error pattern covers a higher order Reed-Muller word is too large.

Observe that the right-hand side of (3.1) never exceeds $n/2$ as should be expected for binary codes. The following corollary shows that we approach $n/2$ asymptotically.

Corollary 3.1 *Consider the codes $\mathcal{RM}(r,m)$ for fixed r and let λ be constant (or sub-exponential in m). Denote by τ_m the maximum number of errors that we can correct by Algorithm 3.1 for the particular value of m and λ as in (3.5). Define the maximum error rate $\varepsilon_m = \tau_m/2^m$. If $\pi_1 \leq 2^{k_{r+\rho}-2^m+\tau}$, then*

$$\lim_{m\to\infty} \varepsilon_m = \frac{1}{2} \ .$$

Proof: Choose $\rho = \lceil \frac{m-2}{2} \rceil - r$. We show that in this case each of the two expressions in $\min\{2^m - k_{r+\rho} - \lambda, k_\rho - 1\}$ normalized by 2^m converges to $1/2$ when taking m to infinity. The first is

$$2^m - \sum_{i=0}^{r+\rho} \binom{m}{i} - \lambda = 2^m - \sum_{i=0}^{\lceil \frac{m-2}{2} \rceil} \binom{m}{i} - \lambda \geq 2^{m-1} - \lambda$$

and the second expression is

$$\sum_{i=0}^{\rho} \binom{m}{i} - 1 \geq \sum_{i=0}^{\lceil \frac{m}{2} \rceil} \binom{m}{i} - \sum_{i=\lceil \frac{m-2}{2} \rceil - r + 1}^{\lceil \frac{m}{2} \rceil} \binom{m}{i} \geq 2^{m-1} - r\binom{m}{\lceil \frac{m}{2} \rceil} \ .$$

$r \diagdown m$	5		6		7		8		9		10	
1	8	**6**	16	**12**	32	**28**	64	**83**	128	**129**	256	**376**
2	4		8	**6**	16	**19**	32	**36**	64	**120**	128	**175**
3	2		4		8	**7**	16	**27**	32	**45**	64	**166**
4	1		2		4		8	**8**	16	**36**	32	**55**
5			1		2		4		8	**9**	16	**46**
6					1		2		4		8	**10**
7							1		2		4	**1**
8									1		2	
9											1	

$r \diagdown m$	11		12	
1	512	**561**	1024	**1576**
2	256	**552**	512	**793**
3	128	**231**	256	**784**
4	64	**222**	128	**298**
5	32	**66**	64	**289**
6	16	**57**	32	**78**
7	8	**11**	16	**69**
8	4	**2**	8	**12**
9	2		4	**3**

Table 3.1: Half the minimum distance of Reed-Muller codes 2^{m-r-1} verse the estimated error correcting capacity τ in (3.5) for $\lambda = 10$.

Both lower bounds converge to $1/2$ when normalizing by 2^m and $m \to \infty$, the latter by the estimate on binomial coefficients (16) in (MacWilliams and Sloane 1977, Chapter 10). Since one of the values is always below $1/2$ for the different ρ, the result follows. □

Corollary 3.1 shows that in this setting the decoder is asymptotically capable of correcting up to the maximum number of errors possible.

3.6 Complexity of the Algorithm

We would like to estimate the running time of Algorithm 3.1 as a function in the block length $n = 2^m$ of the Reed-Muller code $\mathcal{RM}(r, m)$. In the following we will assume that solving a system of linear equations takes time $O(a^3)$ with a being the maximum of the number of unknowns and the number of equations (using for example Gaussian elimination).

The complexity of *Step 1* is negligible and can be done once and for all.

A straight-forward implementation of *Step 2* which finds the boolean polynomial Q, would repeatedly solve a homogeneous system of equations for increasing order of Q_1. The unknowns are the coefficients of Q_1 and Q_2 which are chosen as linear combinations of the p_j and boolean monomials of degree $\leq r + \rho$ respectively. This gives a complexity of $O(n^4)$. However, as increasing the order of Q_1 by 1 simply involves adding another equation and another

unknown, a more efficient approach would be to solve the progressively larger system of equations by reusing old knowledge. This gives a complexity of $O(n^3)$ for Step 2. Approaches that exploit the structure of the system of equations might lower this complexity. In Sudan's algorithm for Reed-Solomon codes such an approach is possible where an $O(sn^2)$ interpolation algorithm is described for low values of s (Nielsen and Høholdt 1999).

Step 3 should find $f \in R_{[x_1,\ldots,x_m]}$ such that $\deg f \leq r$, $f(P_i) = y_i$ for those i where $Q_1(P_i) = 1$. In particular if $Q_1 = 1$, then $Q_2(P_i) = y_i$ for all i. This is desirable to happen when no errors occurred such that $Q = y + f$ where f corresponds to the sent codeword. Therefore the choice $p_0 = 1$ is advantageous. In general Step 3 defines a system of linear equations and under the conditions of Theorem 3.2 there is a solution. Finding one of those as well as a basis of the solution space of the corresponding homogeneous system of linear equations will then represent all solutions to Step 3. Let $\mathbf{q}_1$ denote the $\mathcal{RM}(\rho, m)$ word that corresponds to Q_1. As each non-zero bit in $\mathbf{q}_1$ results in an equation, the complexity of this step depends on the weight of $\mathbf{q}_1$. In the worst case this gives a complexity of $O(n^3)$. In the best case, the first k_r non-zero bits of $\mathbf{q}_1$ treated correspond to information bits of a $\mathcal{RM}(r, m)$ word. Then k_r equations suffice to compute the k_r unknown coefficients of f. If the evaluation of Q_1 is derived by a fast Fourier-like transform, the overall complexity becomes $O(k_r^3 + n \log n)$. Under all circumstances, however, the complexity of Step 3 is dominated by Step 2.

The complexity of *Step 4* depends on the number of solutions to the equation solved in Step 3. Let ℓ be the dimension of the solution space of the homogeneous system from Step 3. Evaluating every solution of which there are 2^ℓ, in all $n = 2^m$ points using a fast Fourier approach then gives a complexity of $O(2^\ell n \log n)$. The overall running time is now $O(2^\ell n \log n + n^3)$. As this is exponential in ℓ, even moderate values of ℓ can give a high running time. However if there is only one solution in Step 3, i.e. $\ell = 0$, then the running time is cubic in n. There are strong indications that this is often the case if $p_0 = 1$ and $p_1, \ldots, p_{k\rho-1}$ are chosen randomly. The conditions for ℓ to be zero follow.

Proposition 3.2 *Let ℓ be the dimension of the solution space of the corresponding homogenous system of equations in Step 3. Furthermore, let $\mathbf{q}_1$ be the $\mathcal{RM}(\rho, m)$ codeword corresponding to Q_1. The following three cases are then equivalent*

1. *$\ell = 0$, i.e., there is exactly one solution to the equation in Step 3;*
2. *$\mathbf{q}_1$ is not covered by any $\mathcal{RM}(r, m)$ codeword $\neq (1, 1, \ldots, 1)$;*
3. *Q_1 has no factors $\neq 1$ of degree at most r (possibly including Q_1) itself.*

Proof : We show each of the following implications by contradiction.

$1 \Rightarrow 2$**:** Let f be a solution to Step 3, i.e. $(y + f)Q_1 = Q$. Assume that there exists a $\mathcal{RM}(r, m)$ codeword $\mathbf{c} \neq (1, 1, \ldots, 1)$ that covers $\mathbf{q}_1$ and the corresponding boolean polynomial $f_\mathbf{c}$ of degree $\leq r$. Then $(f_\mathbf{c} + 1)(P_i) = 0$

when $Q_1(P_i) = 1$. Since $f + f_{\mathbf{c}} + 1$ now evaluates to y_i at those P_i where $Q_1(P_i) = 1$ and is of degree $\leq r$, this implies that $f + f_{\mathbf{c}} + 1$ is also a solution to Step 3. The fact that $\mathbf{c} \neq (1, 1, \ldots, 1)$ implies $f \neq f + f_{\mathbf{c}} + 1$ and we get a contradiction to the assumption.

$2 \Rightarrow 3$**:** Assume that there is a factor $f_{\mathbf{c}} \neq 1$ of Q_1 with $\deg f_{\mathbf{c}} \leq r$. This implies that $(f_{\mathbf{c}}+1)Q_1 = 0$ by Theorem 3.1. Let $\mathbf{c} \neq (1, 1, \ldots, 1)$ be the $\mathcal{RM}(r, m)$ codeword corresponding to $f_{\mathbf{c}}$. Then $(f_{\mathbf{c}} + 1)$ evaluates to 0 at those P_i where $Q_1(P_i) = 1$ and $\mathbf{c} \neq (1, 1, \ldots, 1)$ must cover q_1. This contradicts the conditions.

$3 \Rightarrow 1$**:** Assume that $\ell > 0$, i.e. the equation in Step 3 has two distinct solutions f_1 and f_2. Then by Theorem 3.4 $(y + f_1)Q_1 = Q_2$ and $(y + f_2)Q_1 = Q_2$. Therefore $(f_1 + f_2)Q_1 = 0$, i.e., $f_1 + f_2 + 1 \neq 1$ is a factor of Q by Theorem 3.1 which contradicts our assumption.

□

Proposition 3.2 makes it apparent that 2^ℓ is the number of factors $\neq 1$ in Q_1 of degree at most r and we are able to prove Theorem 3.3.

Proof of Theorem 3.3: Follows directly from Proposition 3.2 and the running time considerations for Step 4. □

Of course the question is, how often does Q_1 have factors of degree $\leq r$? Can we choose Q such that Q_1 has as few factors of that kind as possible? When there are more than one codeword $\mathbf{c}$ closest to the received word $\mathbf{y}$, then it is desired that Q_1 has factors $f \neq 1$ with $\deg f \leq r$. By Theorem 3.4 this will also happen.

In any case there will definitely be multiple solutions if $r \geq \rho$ and $Q_1 \neq 1$, since then Q_1 has itself as a factor of degree ρ. If one uses for example $p_0 = 1$, $p_1 = x_1$, $p_2 = x_2, \ldots, p_{k_\rho - 1} = x_{m-\rho+1} \cdots x_{m-1} x_m$ and there are only few errors, then Q_1 will be of small degree which might be $\leq r$. As we have seen this is not desirable. Using boolean monomials therefore actually results in a counter intuitive situation where the decoder does not work properly if there are too few errors! By injecting additional errors, one can "fix" this.

The nicer approach however is to choose $r < \rho$, $p_0 = 1$ and $p_1, \ldots, p_{k_\rho - 1}$ at random. The boolean polynomial Q_1 will then also be random most likely of degree ρ constrained only by the requirement that $Q_1(P_i) = 0$ for those i that are error positions for some $\mathbf{c}$ closest to $\mathbf{y}$. When choosing the p_j randomly as above in our simulation of Algorithm 3.1 with higher-order Reed-Muller codes of length up to 512, we practically always got only one solution to Step 3 and therefore to Step 4. In other words, a random Q_1 with $\deg Q_1 \leq \rho$ and $Q_1(P_i) = 0$ when i is an error position, seems to have very slim chances of having factors of degree $\leq r$. The correct probability does not seem easy to establish or even to estimate although it appears to be very low. Asking when a completely random pattern covers a codeword or vice versa is easier to answer using arguments like those in Section 3.5. For self-dual codes like $\mathcal{RM}(\frac{m-1}{2}, m)$, m odd, one can even upper-bound the probability that a randomly chosen

codeword covers another randomly chosen codeword by using generalizations of Gleason's theorem to biweight enumerators (MacWilliams et al. 1972). All these probabilities are asymptotically low. This leads us to Conjecture 3.2 which states that the related probability that we seek also goes to zero as the code length increases.

Even if $\ell > 0$ is too large to search through all the solutions for the correct boolean polynomials, one might choose other $p_0, \ldots, p_{k_\rho - 1}$ resulting in another solution to Step 2, but with the same desired factors by Theorem 3.4. Similarly one can just choose another order of the previous p_j. In fact a solution with respect to the original p_j with higher order Q_1 will correspond to solutions to Step 2 with respect to a different ordering of the p_j. Depending on how Step 2 is implemented one might obtain several boolean polynomials Q automatically. A new $Q \in R_{[x_1,\ldots,x_m,y]}$ will result in a new system of equations in Step 3. Considering this new system together with the old one, should hopefully decrease the number of solutions. The procedure can of course be iterated to further lower the number of solutions and this approach works well in practice. Notice that if one can bound the probability π_2 of having many solutions just non-negligibly away from 1, then one might also be able to construct a polynomial-time algorithm which fails with negligible probability by simply repeating Steps 1 and 2 with new orderings in the above manner until the combined system of equations in Step 3 has only the closest codewords as solutions.

Furthermore, the algorithm does not have to process all the points. One can stop as soon as Step 3 has an adequate number of solutions. In a situation with side information, a good approach would be to process the most reliable points first. Erasures are dealt with by simply omitting the points in question from the interpolation.

The memory requirement of the algorithm is $O(n^2)$ for storing the various systems of equations.

3.7 Conclusions

Algorithm 3.1 has been implemented in the computer algebra system Singular (Greuel et al. 1998). Among other codes, it successfully decoded $\mathcal{RM}(2, 9)$ with half the minimum distance being 64 using $\rho = 3$. Due to the known weight distribution of $\mathcal{RM}(2 + 3, 9)$, we know by Fig. 3.1 (b) that the failure probability π_1 for up to 122 random errors is at most 0.01. In each of 10 tries with 120 errors, Step 2 of the algorithm produced only one candidate so no time was wasted filtering away wrong boolean polynomials. The time spent decoding one word was approximately one minute for an unoptimized implementation running on a Pentium 200 MHz under Linux.

Our algorithm improves the results given by Ar et al. (1999) concerning the reconstruction of "noisy multivariate polynomials" in the binary case. It might be possible to extend our method to work also with generalized Reed-Muller codes which have arbitrary finite fields as alphabets. Further Ar et al. (1999) present a more general version of Sudan's algorithm not directly related to coding theory which finds relations $f(x, y) = 0$ that hold with high probabil-

ity. The same sort of generalization ought to be possible in the Reed-Muller setting. It would allow the discovery of probabilistic relations of the type $f'(x_1, \ldots, x_{m_1}, y_1, \ldots, y_{m_2}) = 0$ as opposed to $y = f(x_1, \ldots, x_{m_1})$.

The algorithm can be used to generalize the cryptanalytical results of Jakobsen (1998) and Jakobsen and Knudsen (1999) to a kind of probabilistic interpolation attack for bit-oriented block ciphers. Such work is currently in progress. For these purposes it sometimes suffices to know whether there exists a low-degree approximation without actually obtaining it. In this case, the first two steps of the algorithm are enough. If the resulting boolean polynomial Q has a low degree, then there must exist a low degree approximation. The algorithm might also be applicable to areas like hardware optimization and construction of binary decision trees.

Finally, to verify the conjectures in practice for longer codes of higher order, more simulation results are needed. To achieve this, an optimized, low-level implementation of the algorithm using direct bit manipulations would be useful.

Acknowledgments

The authors wish to thank Tom Høholdt and Jørn Justesen for invaluable discussions.

References

Ar, S., R. J. Lipton, and M. Sudan (1999). Reconstructing Algebraic Functions from Erroneous Data. *SIAM Journal on Computing 28*(2), 487–510.

Berlekamp, E. R. (1996). Bounded Distance +1 Soft-Decision Reed-Solomon Decoding. *IEEE Transactions on Information Theory 42*(3), 704–720.

Greuel, G.-M., G. Pfister, and H. Schönemann (1998). Singular Version 1.2 User Manual. In *Reports on Computer Algebra*, Number 21. Centre for Computer Algebra, University of Kaiserslautern.
`http://www.singular.uni-kl.de`.

Jakobsen, T. (1998). Cryptanalysis of Block Ciphers with Probabilistic Non-linear Relations of Low Degree. In H. Krawczyk (Ed.), *Advances in Cryptology - CRYPTO '98*, Volume 1462 of *Lecture Notes in Computer Science*, pp. 212–222. Springer.

Jakobsen, T. and L. Knudsen (1999). Attacks on Block Ciphers of Low Algebraic Degree. To appear in the *Journal of Cryptology*. Preliminary version: The Interpolation Attack on Block Ciphers. In E. Biham (Ed.) (1997), *Fast Software Encryption*, Volume 1267 of *Lecture Notes in Computer Science*, pp. 212–222. Springer.

Kasami, T., T. Nobuki, and S. Azumi (1976). On the Weight Enumeration of Weights Less than $2.5d$ of Reed-Muller Codes. *Information and Control 30*, 380–395.

MacWilliams, F. J., C. L. Mallows, and N. J. A. Sloane (1972). Generalizations of Gleason's Theorem on Weight Enumerators of Self-Dual Codes. *IEEE Transactions on Information Theory 18*(6), 794–805.

MacWilliams, F. J. and N. J. A. Sloane (1977). *The Theory of Error-Correcting Codes*. North-Holland Mathematical Library.

Menezes, A. J., P. C. van Oorschot, and S. A. Vanstone (1996). *Handbook of Applied Cryptography*. Discrete Mathematics and its Applications. CRC Press.

Muller, D. E. (1954). Application of Boolean Algebra to Switching Circuit Design and to Error Detection. *IEEE Transactions on Computers 3*, 6–12.

Nielsen, R. R. and T. Høholdt (1999). Decoding Reed-Solomon Codes beyond Half the Minimum Distance. In J. Buchmann, T. Høholdt, H. Stichtenoth, and H. Tapia-Recillas (Eds.), *Coding Theory, Cryptography, and Related Areas*. Springer.

Reed, I. S. (1954). A Class of Multiple-Error-Correcting Codes and the Decoding Scheme. *IEEE Transactions on Information Theory 4*, 38–49.

Shimoyama, T. and T. Kaneko (1998). Quadratic Relation of S-box and Its Application to the Linear Attack of Full Round DES. In H. Krawczyk (Ed.), *Advances in Cryptology - CRYPTO '98*, Volume 1462 of *Lecture Notes in Computer Science*, pp. 200–211. Springer.

Shokrollahi, M. A. and H. Wasserman (1999). List Decoding of Algebraic-Geometric Codes. *IEEE Transactions on Information Theory 45*(2), 432–437.

Sudan, M. (1997). Decoding of Reed-Solomon Codes beyond the Error-Correction Bound. *Journal of Complexity 13*, 180–193.

Sugita, T., T. Kasami, and T. Fujiwara (1996). Weight Distributions of the Third and Fifth Order Reed-Muller Codes of Length 512. Technical Report NAIST-IS-TR96006, Nara Institute of Science and Technology. `http://isw3.aist-nara.ac.jp/IS/TechReport2/report/96006.ps`.

Welch, L. R. and E. R. Berlekamp (1986). Error Correction of Algebraic Block Codes. U.S. Patent 4 633 470, issued Dec. 30.

4. GENERALIZED GEOMETRIC GOPPA CODES

Agnes E. Heydtmann

Department of Mathematics, Building 303, Technical University of Denmark,
DK-2800 Kongens Lyngby, Denmark. Agnes.Heydtmann@mat.dtu.dk

Abstract

Conventional geometric Goppa codes are defined in terms of functions of an algebraic function field associated with a divisor evaluated in places of degree 1. The generalization that will be treated here allows evaluations in places of arbitrary degree. With the appropriate inner product, the dual of the code can be defined and described in terms of Weil differentials similarly to conventional geometric Goppa codes. A decoding algorithm is derived.

4.1 Introduction

Geometric Goppa codes have first been described in (Goppa 1981). Since then they have been subject of extensive studies since they were the first and so far only block codes that gave a sequence of codes asymptotically better than the Gilbert-Varshamov bound (Tsfasman et al. 1982; Garcia and Stichtenoth 1995). Recently, Xing et al. (1999) introduced a code based on the generalization of geometric Goppa codes that is discussed in this paper. The generalization is a subspace of the Cartesian product of possibly different extension fields of the finite base field $\mathbb{F}_q$. To obtain a conventional code over $\mathbb{F}_q$, the authors concatenate every entry with an appropriate $\mathbb{F}_q$-code. The result is some examples of optimal codes in the same paper and some codes with improved minimum distance in (Ding et al. 2000) for a given length and dimension in comparison with the tables by Brouwer (1998).

The original motivation for the present work was to find a decoding algorithm for the new codes of Xing et al. (1999). A natural approach seemed to be to develop a method to decode what can be considered as the outer code, since there are known methods to decode concatenated codes that certainly can be applied to this slightly more complicated setting (MacWilliams and Sloane 1977; Ericson 1988). Since the outer code is related to a conventional geometric Goppa code, it was furthermore natural to search for an answer by studying these codes and their decoding algorithm. The standard methods of decoding geometric Goppa codes (Stichtenoth 1993; Farran 2000) require that the dual

code is known such that parity check equations can be obtained. It is for this reason that this paper focuses on the definition and expression of the dual of generalized geometric Goppa codes. Through related investigations it becomes apparent how codewords of the new code can be considered as codewords of the conventional geometric Goppa code associated with the conorm of the divisors involved in the appropriate constant field extension of the underlying algebraic function field. This insight, together with the dual code, make it possible to improve the bound on the minimum distance of the new concatenated codes slightly, and to derive a decoding algorithm over the base field, i.e. it is not necessary to do arithmetic over an extension field.

In Section 4.2, the necessary concepts from algebraic function fields are recalled. The generalization of geometric Goppa codes and their properties are presented in Section 4.3. Section 4.4 shows how to define and represent the dual of a generalized geometric Goppa code. The knowledge that is obtained here is used in Section 4.5 to get a new bound on the minimum distance of the generalized geometric Goppa codes and the concatenated codes. The decoding algorithm is derived in Section 4.6. An example demonstrating the construction of both the generalized code and its dual is given in Section 5.5. Concluding remarks can be found in Section 5.6.

The main results are indeed very similar to the corresponding ones related to the original geometric Goppa codes, but to get there, some results concerning extensions of algebraic function fields are needed. The presentation follows the relevant parts of (Stichtenoth 1993) closely, to make the parallels apparent.

4.2 Preliminaries on Algebraic Function Fields

To start with, some concepts and notation from the theory of algebraic function fields need introducing. Most notation in the following is similar to that in (Stichtenoth 1993), which should also be considered for a thorough introduction to the topic.

Let $L \supseteq K$ be an *algebraic function field*. Denote by $\mathbb{P}_L$ its set of *places* and v_P the *discrete valuation* of a place $P \in \mathbb{P}_L$. Let A be a *divisor* of L, i.e. a formal sum of places. Its *degree* is denoted by $\deg A$ and its *support* by $\operatorname{supp} A$. Also, let B be a divisor. When $v_P(B) \geq v_P(A)$ for all $P \in \mathbb{P}_L$, then $B \geq A$ is written. Let (f) denote the *principal divisor* of the function $0 \neq f \in L$. Define further the K-vector space of functions associated with the divisor A

$$\mathcal{L}(A) = \{f \in L \mid (f) \geq -A\} \cup \{0\}$$

which is of central importance for the construction of geometric Goppa codes. By the Riemann-Roch Theorem, $\dim \mathcal{L}(A) \geq \deg A - g + 1$, with equality if $\deg A > 2g - 2$. Also, if $\deg A < 0$, then $\dim \mathcal{L}(A) = 0$.

Consider now the *finite algebraic extension* $L' \supseteq K'$ of $L \supseteq K$. Let $\mathrm{Tr}_{K' \subseteq K}$, $\mathrm{Tr}_{L' \supseteq L}$ be the corresponding trace functions and recall the concepts related to Weil differentials: Let $\alpha = (\alpha_P)_{P \in \mathbb{P}_L}$ be an *adele* of L. The *adele space* is denoted by $\mathcal{A}_L$ and

$$\mathcal{A}_{L' \supseteq L} = \{\alpha' \in \mathcal{A}_{L'} \mid \alpha'_{P'} = \alpha'_{Q'} \text{ when } P', Q' \in \mathbb{P}_{L'} : P' \cap L = Q' \cap L\}$$

is a subspace of $\mathcal{A}_{L'}$. The trace function is extended to $\mathrm{Tr}_{L'\supseteq L} : \mathcal{A}_{L'\supseteq L} \to \mathcal{A}_L$ by defining $\mathrm{Tr}_{L'\supseteq L}(\alpha') = \alpha = (\alpha_P)_{P\in\mathbb{P}_L}$ with $\alpha_P = \mathrm{Tr}_{L'\supseteq L}(\alpha_{P'})$ for $\alpha' = (\alpha_{P'})_{P'\in\mathbb{P}_{L'}} \in \mathcal{A}_{L'\supseteq L}$ and $P'|P$. The subspace of the adele space associated with the divisor A is

$$\mathcal{A}_L(A) = \{\alpha \in \mathcal{A}_L \mid v_P(\alpha) \geq -v_P(A),\, \forall\, P \in \mathbb{P}_L\}.$$

The *principal adele* of $f \in L$ is the adele with all components equal to f. The field L is embedded into $\mathcal{A}_L$ in this way. The L-vector space of *Weil differentials* of $L \supseteq K$ is denoted by Ω_L. Let (ω) denote the *canonical divisor* associated with ω. The subspace over K associated with the divisor A is

$$\Omega_L(A) = \{\omega \in \Omega_L \mid \omega \text{ vanishes on } \mathcal{A}_L(A)+L\} = \{\omega \in \Omega_L \mid \omega = 0 \text{ or } (\omega) \geq A\}.$$

Let $i(A)$ denote the *index of speciality* of A. Then $\dim \Omega_L(A) = i(A)$. Further, let $\iota_P : L \hookrightarrow \mathcal{A}_L$ be the embedding such that $\iota_P(f) = (\alpha_Q)_{Q\in\mathbb{P}_L}$ is the adele with $\alpha_P = f$ and $\alpha_Q = 0$ for $Q \neq P$. The *local component* $\omega_P : L \to K$ of a Weil differential ω with respect to the place P is defined by $\omega_P(f) = \omega\big(\iota_P(f)\big)$.

The *conorm* of a divisor A with respect to $L' \supseteq L$ is denoted by $\mathrm{Con}_{L'\supseteq L}(A)$.

Let $\omega \in \Omega_L$ and $L' \supseteq K'$ be a finite separable extension of the function field $L \supseteq K$. Recall that if $K' \supseteq K$ is Galois and $\mathrm{Gal}(K' \supseteq K)$ its Galois group, then $\mathrm{Tr}_{K'\supseteq K} = \sum_{\sigma\in\mathrm{Gal}(K'\supseteq K)} \sigma$. The *cotrace* of ω is the unique Weil differential $\mathrm{Cotr}_{L'\supseteq L}(\omega) \in \Omega_{L'}$ such that

$$\mathrm{Tr}_{K'\supseteq K}\big(\mathrm{Cotr}_{L'\supseteq L}(\omega)(\alpha)\big) = \omega\big(\mathrm{Tr}_{L'\supseteq L}(\alpha)\big)$$

for all $\alpha \in \mathcal{A}_{L'\supseteq L}$. The map $\mathrm{Cotr}_{L'\supseteq L}(\omega)$ can be constructed explicitly. To this extent, the dual space $K^* = \{\varphi : K' \to K \mid \varphi \text{ is } K\text{-linear}\}$ of K' over K as with dimension $[K' : K]$ is defined. Defining further $\lambda \cdot \varphi(\mu) = \varphi(\lambda\mu)$ for $\varphi \in K^*$ and $\lambda, \mu \in K'$ turns K^* into a K'-vector space of dimension 1. Now, $\mathrm{Tr}_{K'\supseteq K}$ is certainly a non-trivial element of K^* and thus each $\varphi \in K^*$ has a representation $\varphi = \lambda \cdot \mathrm{Tr}_{K'\supseteq K}$ for some $\lambda \in K'$. The following lemma describes the construction of the cotrace.

Lemma 4.1 *Let $L' \supseteq K'$ be a finite separable extension of the function field $L \supseteq K$ and consider $\omega \in \Omega_L$. Define $\omega_1 = \omega \circ \mathrm{Tr}_{L'\supseteq L} : \mathcal{A}_{L'\supseteq L} \to K$ and $\omega_2 : \mathcal{A}_{L'} \to K$ such that $\omega_2(\alpha') = \omega_1(\alpha)$ where $\mathcal{A}_{L'} \ni \alpha' = \alpha + \beta$ such that $\alpha \in \mathcal{A}_{L'\supseteq L}$ and $\beta \in \mathcal{A}_{L'}\big(\mathrm{Con}_{L'\supseteq L}\big((\omega)\big)\big)$. For $\alpha' \in \mathcal{A}_{L'}$ define $\varphi_{\alpha'} \in K^*$ by $\varphi_{\alpha'}(\mu) = \omega_2(\mu\alpha')$ where $\mu \in K'$. Let further $\lambda_{\alpha'} \in K'$ such that $\varphi_{\alpha'} = \lambda_{\alpha'} \cdot \mathrm{Tr}_{K'\supseteq K}$. Then*

$$\mathrm{Cotr}_{L'\supseteq L}(\omega)(\alpha') = \lambda_{\alpha'}.$$

Refer to (Stichtenoth 1993, III.4.6 and III.4.9) for the proof.

Let $K' \supseteq K$ be a finite Galois extension. The product $L' = K'L$ of the two fields K' and L is the smallest field containing both fields. The function field $L' \supseteq K'$ is called a *constant field extension* of $L \supseteq K$. It is finite separable and Galois since $K' \supseteq K$ is. Also, $\mathrm{Tr}_{L'\supseteq L}|_{K'} = \mathrm{Tr}_{K'\supseteq K}$ and if $\sigma \in \mathrm{Gal}(L' \supseteq L)$, then $\sigma|_{K'} \in \mathrm{Gal}(K' \supseteq K)$.

For x a separating element of L, the *exact differential* in the *differential module* is denoted by dx. Let P_∞ be the unique *infinite place* of the *rational function field* $K(x) \subseteq K$ and η the unique Weil differential of the function field with $(\eta) = -2P_\infty$ and $\eta_{P_\infty}(x^{-1}) = -1$ (Stichtenoth 1993, I.7.4). By (Stichtenoth 1993, IV.3.2 (c)), the exact differentials dx stand in one-to-one correspondence with the Weil differentials $\mathrm{Cotr}_{L \supseteq K(x)}(\eta) \in \Omega_L$ and are therefore identified with them (Stichtenoth 1993, IV.3.7).

Let $P \in \mathbb{P}_L$ be a place of degree one and $\omega \in \Omega_L$. The *residue* of ω at P is denoted by $\mathrm{res}_P(w)$.

4.3 The Generalization

In this section, the generalized geometric Goppa code is introduced, which the codes of Xing et al. (1999) are based on. Their construction is a concatenation of the generalized geometric Goppa code presented here with possibly various different codes.

Let $\mathbb{F}_q$ be a finite field with q elements, $\mathbb{N} = \{1, 2, 3, \ldots\}$ and $k_1, \ldots, k_n \in \mathbb{N}$. Denote the Cartesian product of the fields $\mathbb{F}_{q^{k_i}}$ by $\prod_{i=1}^n \mathbb{F}_{q^{k_i}}$. In this paper, the standard definition of a code is extended:

Definition 4.1

1. *A* code C *over* $\mathbb{F}_q$ *is a subspace of the* $\mathbb{F}_q$*-linear space* $\prod_{i=1}^n \mathbb{F}_{q^{k_i}}$.
2. *The* length *of the code is the number* n.
3. *The* dimension k *of the code is the dimension of* C *as an* $\mathbb{F}_q$*-vector space.*

Let $\mathbf{a} = (a_1, \ldots, a_n)$, $\mathbf{b} = (b_1, \ldots, b_n) \in \prod_{i=1}^n \mathbb{F}_{q^{k_i}}$, $k_i, n \in \mathbb{N}$.

4. *The* Hamming distance *of* $\mathbf{a}$ *and* $\mathbf{b}$ *is defined as* $\mathrm{d}(\mathbf{a}, \mathbf{b}) = |\{i \mid a_i \neq b_i\}|$.
5. *The* weight *of* $\mathbf{a}$ *is* $\mathrm{w}(\mathbf{a}) = \mathrm{d}(\mathbf{a}, 0) = |\{i \mid a_i \neq 0\}|$.
6. *The* minimum distance d *of* $C \neq 0$ *is defined as* $d = \min\{\mathrm{d}(\mathbf{a}, \mathbf{b}) \mid \mathbf{a}, \mathbf{b} \in C, \mathbf{a} \neq \mathbf{b}\} = \min\{\mathrm{w}(\mathbf{a}) \mid 0 \neq \mathbf{a} \in C\}$.
7. *An* $[n, k, d]$ *code is one of length* n*, dimension* k *and minimum distance* d.

Note that in general C is a code over a mixed alphabet and that possibly $k > n$.

Some notation is fixed for the remainder of this paper:

$F \supseteq \mathbb{F}_q$ is an algebraic function field of genus g,
$D = P_1 + \cdots + P_n$ with P_i pairwise distinct places of $F \supseteq \mathbb{F}_q$ of degree k_i,
$N = \sum_{i=1}^n k_i = \dim \prod_{i=1}^n \mathbb{F}_{q^{k_i}} = \deg D$, and
G is a divisor of $F \supseteq \mathbb{F}_q$ such that $\mathrm{supp}\, G \cap \mathrm{supp}\, D = \emptyset$

Now the extended class of geometric Goppa codes can be defined.

Definition 4.2 *The* generalized geometric Goppa code *associated with the divisors D and G is the $\mathbb{F}_q$-vector space*

$$C_{\mathcal{L}}(D,G) = \big\{ \big(f(P_1),\ldots,f(P_n)\big) \mid f \in \mathcal{L}(G) \big\} \subseteq \prod_{i=1}^{n} \mathbb{F}_{q^{k_i}}.$$

Naturally, the parameters of this new code are of interest. In the particular case where $k_1 = \cdots = k_n = 1$ the following proposition simplifies to (Stichtenoth 1993, II.2.2).

Proposition 4.1 *The space $C_{\mathcal{L}}(D,G)$ is an $[n, k_{\mathcal{L}}, d_{\mathcal{L}}]$ code with parameters*

$$k_{\mathcal{L}} = \dim \mathcal{L}(G) - \dim \mathcal{L}(G-D) \qquad \text{and} \qquad d_{\mathcal{L}} \geq n - \deg G.$$

Further if $\deg G < N$, then $k_{\mathcal{L}} = \dim \mathcal{L}(G) \geq \deg G - g + 1$ and consequently if $2g-2 < \deg G < N$, then $k_{\mathcal{L}} = \deg G - g + 1$.

Proof: Consider the evaluation map

$$\begin{array}{rccc} \mathrm{ev}_D: & \mathcal{L}(G) & \to & C_{\mathcal{L}}(D,G) \\ & f & \mapsto & \big(f(P_1),\ldots,f(P_n)\big) \end{array}$$

which is $\mathbb{F}_q$-linear and certainly surjective with kernel

$$\begin{aligned} \mathrm{Ker}(\mathrm{ev}_D) &= \{f \in \mathcal{L}(G) \mid f(P_i) = 0 \text{ for } i = 1,\ldots,n\} \\ &= \{f \in \mathcal{L}(G) \mid v_{P_i}(f) > 0 \text{ for } i = 1,\ldots,n\} \\ &= \mathcal{L}(G-D). \end{aligned}$$

Thus $\dim C_{\mathcal{L}}(D,G) = \dim \mathcal{L}(G) - \dim \mathcal{L}(G-D)$. If $\deg G < N$, then $\deg(G-D) < 0$ and $\dim \mathcal{L}(G-D) = 0$. Therefore $k_{\mathcal{L}} = \dim \mathcal{L}(G) \geq \deg G - g + 1$ and $k_{\mathcal{L}} = \deg G - g + 1$ if $2g-2 < \deg G < N$.

Regarding the minimum distance, take $f \in \mathcal{L}(G)$ such that $\mathrm{w}\big(\mathrm{ev}_D(f)\big) = d_{\mathcal{L}}$. Let $P_{i_1},\ldots,P_{i_{n-d_{\mathcal{L}}}}$ be those places such that $f(P_{i_j}) = 0$, $j = 1,\ldots,n-d_{\mathcal{L}}$. Then

$$0 \neq f \in \mathcal{L}(G - P_{i_1} - \cdots - P_{i_{n-d_{\mathcal{L}}}}),$$

implying

$$0 \leq \deg(G - P_{i_1} - \cdots - P_{i_{n-d_{\mathcal{L}}}}) = \deg G - \sum_{j=1}^{n-d_{\mathcal{L}}} k_{i_j} = \deg G - \sum_{j=1}^{n-d_{\mathcal{L}}} (k_{i_j} - 1) - n + d_{\mathcal{L}}$$

and $d_{\mathcal{L}} \geq n - \deg G$. □

The new codes by Xing et al. (1999) are then constructed as follows:

Definition 4.3 *Let $C_1,\ldots,C_n$ be $[n_i, k_i, d_i]$ codes, $1 \leq i \leq n$. For each C_i there is an $\mathbb{F}_q$-linear isomorphism $\pi_i : \mathbb{F}_{q^{k_i}} \to C_i$. The* concatenated generalized geometric Goppa code *is*

$$C_{\mathcal{L}}\big(D, G, C_1,\ldots,C_n\big) = \big\{ \big(\pi_1(c_1),\ldots,\pi_n(c_n)\big) \mid (c_1,\ldots,c_n) \in C_{\mathcal{L}}(D,G) \big\}.$$

If $\deg G < \deg D$, then $C_{\mathcal{L}}(D, G, C_1, \ldots, C_n)$ is an $\mathbb{F}_q$-linear code with parameters $[n', k', d']$ where $n' = \sum_{i=1}^{r} n_i$, $k' \geq \deg G + 1 - g$, equality holding when also $\deg G > 2g - 2$, and

$$d' \geq \sum_{i=1}^{n} d_i - \deg G - \max\left\{ \sum_{i \in I} (d_i - k_i) \,\middle|\, I \subseteq \{1, \ldots, n\} \right\}$$

(Xing et al. 1999, Theorem 3.2). Note that $d' \geq \sum_{i=1}^{n} d_i - \deg G$ when $k_i \geq d_i$. When $\deg G < \deg D$, then $\dim C_{\mathcal{L}}(D, G) = \dim C_{\mathcal{L}}(D, G, C_1, \ldots, C_n)$.

4.4 The Dual Code

It is known that for $k_1 = \cdots = k_n = 1$ there is, apart from the conventional geometric Goppa code $C_{\mathcal{L}}(D, G)$, the code $C_{\Omega}(D, G)$ associated with the divisors D and G. It is defined in terms of the evaluation of local components of Weil differentials. This code is also a geometric Goppa code and the dual of $C_{\mathcal{L}}(D, G)$. In the following a code $C_{\Omega}(D, G)$ will be defined similarly without the restrictions on the degrees k_i. However, Weil differentials of the algebraic function field $F \supseteq \mathbb{F}_q$ and their local components are mappings into the base field $\mathbb{F}_q$. It is natural to require that also $C_{\Omega}(D, G)$ be a code over a mixed alphabet, i.e. an $\mathbb{F}_q$-linear subspace of $\prod_{i=1}^{n} \mathbb{F}_{q^{k_i}}$. The cotrace of a Weil differential for a given finite separable extension will therefore play an important role. Before $C_{\Omega}(D, G)$ can be defined, some more notation related to various field extensions that are considered needs introducing.

Let $\mathbb{F}_{q^\ell}$ be the smallest field that contains all $\mathbb{F}_{q^{k_i}}$ ($\ell = \operatorname{lcm}(k_1, \ldots, k_n)$). Consider the constant field extension $F' = \mathbb{F}_{q^\ell} F \supseteq \mathbb{F}_{q^\ell}$ and define the conorms $D' = \operatorname{Con}_{F' \supseteq F}(D)$, $G' = \operatorname{Con}_{F' \supseteq F}(G)$. The $\mathbb{F}_{q^\ell}$-vector space $C_{\mathcal{L}}(D', G') \subseteq \mathbb{F}_{q^\ell}^N$ is the conventional $[N, k_{\mathcal{L}}, \geq N - \deg G]$ geometric Goppa code associated with the divisors D' and G' of $F' \supseteq \mathbb{F}_{q^\ell}$. Also the constant field extensions $F_{k_i} = \mathbb{F}_{q^{k_i}} F \supseteq \mathbb{F}_{q^{k_i}}$ are needed as well as the corresponding conorms $D_{k_i} = \operatorname{Con}_{F_{k_i} \supseteq F}(D)$ and $G_{k_i} = \operatorname{Con}_{F_{k_i} \supseteq F}(G)$ for $i = 1, \ldots, n$. Observe that $F' \supseteq F_{k_i} \supseteq F$.

Let P_i^* be a fixed place of $F_{k_i} \supseteq \mathbb{F}_{q^{k_i}}$ above P_i for all i and let $F_{P_i}, F_{k_i P_i^*}$ denote the residue class fields of P_i, P_i^* respectively. Since $\deg P_i^* = 1$, $F_{P_i} \cong \mathbb{F}_{q^{k_i}} \cong F_{k_i P_i^*}$. When Q_i^* is an arbitrary place in F_{k_i} above P_i, and Q_i' the only place in F' above Q_i^*, denote by $F_{k_i Q_i^*}, F'_{Q_i'}$ the residue class fields of Q_i^*, Q_i' respectively. The fields $F_{P_i}, F_{k_i Q_i^*}$ can be considered as subfields of $F'_{Q_i'}$. Specifically, P_i' denotes the only place of $F' \supseteq \mathbb{F}_{q^\ell}$ above P_i^*. Throughout the text we identify $f(P_i) = f(P_i^*) = f(P_i') \in \mathbb{F}_{q^{k_i}}$ as well as $f(Q_i^*) = f(Q_i') \in \mathbb{F}_{q^{k_i}}$ for $f \in \mathcal{L}(G) \subseteq \mathcal{L}(G_{k_i}) \subseteq \mathcal{L}(G')$.

Certainly the extensions $F' \supseteq \mathbb{F}_{q^\ell}$, $F_{k_i} \supseteq \mathbb{F}_{q^{k_i}}$ of $F \supseteq \mathbb{F}_q$ are finite separable and Galois since $\mathbb{F}_{q^\ell} \supseteq \mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q$ are Galois extensions.

The next code associated with D and G can now be defined.

Definition 4.4

$$C_{\Omega}(D, G) = \left\{ \left(\operatorname{Cotr}_{F_{k_1} \supseteq F}(\omega)_{P_1^*}(1), \ldots, \operatorname{Cotr}_{F_{k_n} \supseteq F}(\omega)_{P_n^*}(1) \right) \mid \omega \in \Omega_F(G - D) \right\}$$

Similarly to $C_{\mathcal{L}}(D,G)$, $C_\Omega(D,G)$ is a subspace of $\prod_{i=1}^n \mathbb{F}_{q^{k_i}}$ over $\mathbb{F}_q$. The following statement gives the parameters of this new code. Again, the results are an analogue of (Stichtenoth 1993, II.2.7).

Proposition 4.2 *The space $C_\Omega(D,G)$ is an $[n,k_\Omega,d_\Omega]$ code with parameters*

$$k_\Omega = i(G-D) - i(G) \qquad \text{and} \qquad d_\Omega \geq \deg G - (N-n) - (2g-2).$$

Further if $\deg G > 2g-2$, then $k_\Omega = i(G-D) \geq N - \deg G + g - 1$ and consequently if $2g-2 < \deg G < N$, then $k_\Omega = N + g - 1 - \deg G$.

Proof: Consider the $\mathbb{F}_q$-linear mapping

$$\begin{aligned} \rho_D : \ \Omega_F(G-D) &\to C_\Omega(D,G) \\ \omega &\mapsto \left(\mathrm{Cotr}_{F_{k_1}\supseteq F}(\omega)_{P_1^*}(1), \ldots, \mathrm{Cotr}_{F_{k_n}\supseteq F}(\omega)_{P_n^*}(1)\right), \end{aligned}$$

which certainly is surjective. Observe that $v_{P_i}(\omega) \geq -1$ holds for $\omega \in \Omega_F(G-D)$ and further, since $\mathrm{Con}_{F_{k_i}\supseteq F}\big((\omega)\big) = \big(\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)\big)$ in a constant field extension (Stichtenoth 1993, III.4.6,III.5.1 (b)), also $v_{P_i^*}\big(\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)\big) \geq -1$. By (Stichtenoth 1993, II (2.7)),

$$\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)_{P_i^*}(1) = 0 \quad \Longleftrightarrow \quad v_{P_i}(\omega) = v_{P_i^*}\big(\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)\big) \geq 0,$$

and therefore the kernel of the mapping ρ_D is $\Omega_F(G)$. Thus

$$k_\Omega = \dim \Omega_F(G-D) - \dim \Omega_F(G) = i(G-D) - i(G).$$

If $\deg G > 2g-2$, then $i(G) = 0$. Therefore $k_\Omega = i(G-D) = \dim \mathcal{L}(G-D) - \deg(G-D) + g - 1 \geq N - \deg G + g - 1$ and if $2g-2 < \deg G < N$, then $k_\Omega = N - \deg G + g - 1$.

Let $\omega \in \Omega_F(G-D)$ be such that $\mathrm{w}\big(\rho_D(\omega)\big) = d_\Omega$. Let $P_{i_1}, \ldots, P_{i_{n-d_\Omega}}$ be those places such that $\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)_{P_{i_j}^*}(1) = 0$, $j = 1, \ldots, n-d_\Omega$. Then

$$0 \neq \omega \in \Omega_F(G - D + P_{i_1} + \cdots + P_{i_{n-d_\Omega}}),$$

implying

$$\begin{aligned} 2g-2 &\geq \deg(G - D + P_{i_1} + \cdots + P_{i_{n-d_\Omega}}) \\ &= \deg G - N + \sum_{j=1}^{n-d_\Omega} k_{i_j} = \deg G - N + \sum_{j=1}^{n-d} (k_{i_j} - 1) + n - d_\Omega \end{aligned}$$

and $d_\Omega \geq \deg G - (N-n) - (2g-2)$. □

Naturally, concatenating these codes is also possible.

Definition 4.5 *Let $C_1, \ldots, C_n$ be $[n_i, k_i, d_i]$ codes, $1 \leq i \leq n$. For each C_i there is an $\mathbb{F}_q$-linear isomorphism $\pi_i : \mathbb{F}_{q^{k_i}} \to C_i$. Define*

$$C_\Omega\big(D, G, C_1, \ldots, C_n\big) = \big\{\big(\pi_1(c_1), \ldots, \pi_n(c_n)\big) \mid (c_1, \ldots, c_n) \in C_\Omega(D,G)\big\}.$$

If $\deg G > 2g-2$, then by similar reasoning as in (Xing et al. 1999, Theorem 3.2) $C_\Omega(D, G, C_1, \ldots, C_n)$ is an $\mathbb{F}_q$-linear code with parameters $[n', k', d']$ where $n' = \sum_{i=1}^{r} n_i$, $k' \geq N - \deg G + g - 1$, equality holding when also $\deg G < \deg D$, and

$$d' \geq \deg G - (2g-2) - \sum_{i=1}^{n}(k_i - d_i) - \max\left\{ \sum_{i \in I}(d_i - k_i) \,\middle|\, I \subseteq \{1, \ldots, n\} \right\}.$$

Note that $d' \geq \deg G - (2g-2) - \sum_{i=1}^{n}(k_i - d_i)$ when $k_i \geq d_i$. Certainly, when $\deg G > 2g-2$, then $\dim C_\Omega(D, G) = \dim C_\Omega(D, G, C_1, \ldots, C_n)$.

In order to relate $C_\mathcal{L}(D, G)$ and $C_\Omega(D, G)$ to each other, the notion of the dual of a code $C \subseteq \prod_{i=1}^{n} \mathbb{F}_{q^{k_i}}$ will now be introduced.

Definition 4.6

1. *Let* $\langle \cdot, \cdot \rangle : \prod_{i=1}^{n} \mathbb{F}_{q^{k_i}} \times \prod_{i=1}^{n} \mathbb{F}_{q^{k_i}} \to \mathbb{F}_q$ *be defined by*

$$\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i=1}^{n} \mathrm{Tr}_{\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q}(a_i b_i), \quad \mathbf{a} = (a_1, \ldots, a_n),\ \mathbf{b} = (b_1, \ldots, b_n) \in \prod_{i=1}^{n} \mathbb{F}_{q^{k_i}}.$$

   Due to the properties of the trace function, the mapping $\langle \cdot, \cdot \rangle$ *is an inner product.*

2. *Let* $C \subseteq \prod_{i=1}^{n} \mathbb{F}_{q^{k_i}}$ *be a code. Its dual code* $C^\perp$ *is defined as the annihilator with respect to the above inner product:*

$$C^\perp = \left\{ \mathbf{a} \in \prod_{i=1}^{n} \mathbb{F}_{q^{k_i}} \;\middle|\; \langle \mathbf{a}, \mathbf{c} \rangle = 0 \ \forall\, \mathbf{c} \in C \right\}$$

Note that certainly $\dim C^\perp = N - \dim C$. The main theorem of this paper can now be stated.

Theorem 4.1

$$C_\mathcal{L}(D, G)^\perp = C_\Omega(D, G)$$

To prove the claim several lemmas concerned with function field extensions and properties of the cotrace of a Weil differential are needed. The main ingredient of the proof is that elements of $C_\mathcal{L}(D, G)$ and $C_\Omega(D, G)$ corresponding to the function $f \in \mathcal{L}(G)$ and the Weil differential $\omega \in \Omega(G - D)$ respectively can be "blown up" to elements corresponding to f and $\mathrm{Cotr}_{F' \supseteq F}(\omega)$ of the conventional geometric Goppa codes $C_\mathcal{L}(D', G')$ and $C_\Omega(D', G')$ respectively. Elements of $C_\mathcal{L}(D', G')$ will be denoted by $\big(f'(Q_i') \mid Q_i' | P_i, 1 \leq i \leq n\big)$ for some $f' \in \mathcal{L}(G')$ with a fixed order on the Q_i' and those of $C_\Omega(D', G')$ by $(\omega'_{Q_i'}(1) \mid Q_i' | P_i,\ 1 \leq i \leq n)$ for some $\omega' \in \Omega(G' - D')$ correspondingly.

Lemma 4.2

1. *Let* $\sigma_1, \ldots, \sigma_{k_i}$ *be the distinct elements of* $\mathrm{Gal}(F_{k_i} \supseteq F)$. *For any* i, *the sets* $\sigma_1(P_i^*), \ldots, \sigma_{k_i}(P_i^*)$ *are the distinct places of* $F_{k_i} \supseteq \mathbb{F}_{q^{k_i}}$ *above* P_i.

2. *Let $\omega \in \Omega_F$, $f \in F$. Then*

$$\sigma\big(\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)_{P_i^*}(f)\big) = \mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)_{\sigma(P_i^*)}(f)$$

for any $\sigma \in \mathrm{Gal}(F_{k_i} \supseteq F)$

3. *Let $L \supseteq K$ be an algebraic function field and $L' \supseteq K'$ a finite separable extension. Further, let $\omega \in \Omega_L$ and P be a place of L. Then*

$$\sum_{P'|P} \mathrm{Cotr}_{L'\supseteq L}(\omega)_{P'}(f) = \omega_P(f)$$

for $f \in L$.

Proof :

1. Note that $\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q$ is a Galois extension and therefore certainly $F_{k_i} \supseteq F$ is. The rest follows immediately from (Stichtenoth 1993, III.7.1).
2. To prove the claim, the construction of $\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)$ in Lemma 4.1 is used.
   For $\mu \in \mathbb{F}_{q^{k_i}}$ define $\alpha_\mu^* = (\alpha_{\mu Q^*}^*) = \iota_{P_i^*}(\mu f) \in \mathcal{A}_{F_{k_i}}$.
   By the Approximation Theorem (Stichtenoth 1993, I.3.1), there exists $x_{P_i} \in F_{k_i}$ such that

$$v_{Q_i^*}(\alpha_{\mu Q_i^*}^* - x_{P_i}) \geq -v_{Q_i^*}\big(\mathrm{Con}_{F_{k_i}\supseteq F}\big((\omega)\big)\big) \qquad \forall\, Q_i^*|P_i.$$

   Now let $\beta_\mu^* = (\beta_{\mu Q^*}^*) = \sum_{Q_i^*|P_i} \iota_{Q_i^*}(x_{P_i})$. Then $\beta_\mu^* \in \mathcal{A}_{F_{k_i}\supseteq F}$ and $\alpha_\mu^* - \beta_\mu^* \in \mathcal{A}_{F_{k_i}}\big(\mathrm{Con}_{F_{k_i}\supseteq F}\big((\omega)\big)\big)$ by the definition of this space. Thus $\varphi_{\alpha_1^*}(\mu) = \omega_2(\mu\alpha_1^*) = \omega_1(\beta_\mu^*)$.
   Fix $\sigma \in \mathrm{Gal}(F_{k_i} \supseteq F)$ and observe now that

$$\begin{aligned} v_{\sigma(Q_i^*)}\big(\sigma(\alpha_{\mu Q_i^*}^*) - \sigma(x_{P_i})\big) &= v_{Q_i^*}\big(\sigma^{-1}\big(\sigma(\alpha_{\mu Q_i^*}^*) - \sigma(x_{P_i})\big)\big) \\ &= v_{Q_i^*}(\alpha_{\mu Q_i^*}^* - x_{P_i}) \\ &\geq -v_{Q_i^*}\big(\mathrm{Con}_{F_{k_i}\supseteq F}\big((\omega)\big)\big) \\ &= -v_{\sigma(Q_i^*)}\big(\mathrm{Con}_{F_{k_i}\supseteq F}\big((\omega)\big)\big) \end{aligned}$$

   for $Q_i^*|P_i$ by (Stichtenoth 1993, III.5.2 (a)) and the fact that a constant field extension is unramified.
   Let $\nu = \sigma(\mu)$ and define $\alpha_\nu^\diamond = (\alpha_{\nu Q^*}^\diamond) = \iota_{\sigma(P_i^*)}(\nu f) = \iota_{\sigma(P_i^*)}\big(\sigma(\mu f)\big)$. By the above,

$$\begin{aligned} v_{Q_i^*}\big(\alpha_{\nu Q_i^*}^\diamond - \sigma(x_{P_i})\big) &= v_{Q_i^*}\big(\sigma(\alpha_{\mu\sigma^{-1}(Q_i^*)}^*) - \sigma(x_{P_i})\big) \\ &\geq -v_{Q_i^*}\big(\mathrm{Con}_{F_{k_i}\supseteq F}\big((\omega)\big)\big) \end{aligned}$$

   for $Q_i^*|P_i$. Let $\beta_\nu^\diamond = (\beta_{\nu Q^*}^\diamond) = \sum_{Q_i^*|P_i} \iota_{Q_i^*}\big(\sigma(x_{P_i})\big)$. Then $\beta_\nu^\diamond \in \mathcal{A}_{F_{k_i}\supseteq F}$ and $\alpha_\nu^\diamond - \beta_\nu^\diamond \in \mathcal{A}_{F_{k_i}}\big(\mathrm{Con}_{F_{k_i}\supseteq F}\big((\omega)\big)\big)$, i.e.

$$\begin{aligned} \varphi_{\alpha_1^\diamond}(\nu) &= \omega_2(\nu\alpha_1^\diamond) = \omega_1(\beta_\nu^\diamond) = \omega\big(\mathrm{Tr}_{F_{k_i}\supseteq F}(\beta_\nu^\diamond)\big) = \omega_{P_i}\big(\mathrm{Tr}_{F_{k_i}\supseteq F}\big(\sigma(x_{P_i})\big)\big) \\ &= \omega_{P_i}\Big(\sum_{\sigma'\in\mathrm{Gal}(F_{k_i}\supseteq F)} \sigma'\circ\sigma(x_{P_i})\Big) = \omega_{P_i}\Big(\sum_{\sigma'\in\mathrm{Gal}(F_{k_i}\supseteq F)} \sigma'(x_{P_i})\Big) \\ &= \omega_{P_i}\big(\mathrm{Tr}_{F_{k_i}\supseteq F}(x_{P_i})\big) = \omega_1(\beta_\mu^*) = \omega_2(\mu\alpha_1^*) = \varphi_{\alpha_1^*}(\mu). \end{aligned}$$

Now

$$\begin{aligned}\varphi_{\alpha_1^\circ}(\mu) &= \varphi_{\alpha_1^*}\left(\sigma^{-1}(\mu)\right) = \lambda_{\alpha_1^*}\cdot \mathrm{Tr}_{\mathbb{F}_{q^{k_i}}\supseteq\mathbb{F}_q}\left(\sigma^{-1}(\mu)\right)\\ &= \mathrm{Tr}_{\mathbb{F}_{q^{k_i}}\supseteq\mathbb{F}_q}\left(\lambda_{\alpha_1^*}\sigma^{-1}(\mu)\right) = \sum_{\sigma'\in\mathrm{Gal}(F_{k_i}\supseteq F)}\sigma'\left(\lambda_{\alpha_1^*}\sigma^{-1}(\mu)\right)\\ &= \sum_{\sigma'\in\mathrm{Gal}(F_{k_i}\supseteq F)}\sigma'\circ\sigma^{-1}\left(\sigma(\lambda_{\alpha_1^*})\mu\right) = \sum_{\sigma'\in\mathrm{Gal}(F_{k_i}\supseteq F)}\sigma'\left(\sigma(\lambda_{\alpha_1^*})\mu\right)\\ &= \mathrm{Tr}_{\mathbb{F}_{q^{k_i}}\supseteq\mathbb{F}_q}\left(\sigma(\lambda_{\alpha_1^*})\mu\right) = \sigma(\lambda_{\alpha_1^*})\mathrm{Tr}_{\mathbb{F}_{q^{k_i}}\supseteq\mathbb{F}_q}(\mu).\end{aligned}$$

Thus $\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)_{P_i^*}(f) = \lambda_{\alpha_1^*}$ and $\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)_{\sigma(P_i^*)}(f) = \sigma(\lambda_{\alpha_1^*})$ with the notation of Lemma 4.1. The claim follows.

3. Again Lemma 4.1 is needed.
   Note that

$$\sum_{P'|P}\mathrm{Cotr}_{L'\supseteq L}(\omega)_{P'}(f) = \mathrm{Cotr}_{L'\supseteq L}(\omega)\Big(\sum_{P'|P}\iota_{P'}(f)\Big).$$

   Calculate

$$\begin{aligned}\varphi_{\sum_{P'|P}\iota_{P'}(f)}(\mu) &= \omega_2\Big(\mu\sum_{P'|P}\iota_{P'}(f)\Big) = \omega_1\Big(\sum_{P'|P}\mu\iota_{P'}(f)\Big)\\ &= \omega\circ\mathrm{Tr}_{L'\supseteq L}\Big(\sum_{P'|P}\iota_{P'}(\mu f)\Big) = \omega\left(\iota_P\left(\mathrm{Tr}_{L'\supseteq L}(\mu f)\right)\right)\\ &= \omega\left(\iota_P\left(f\mathrm{Tr}_{K'\supseteq K}(\mu)\right)\right) = \omega\left(\iota_P(f)\mathrm{Tr}_{K'\supseteq K}(\mu)\right)\\ &= \omega\left(\iota_P(f)\right)\cdot\mathrm{Tr}_{K'\supseteq K}(\mu) = \omega_P(f)\cdot\mathrm{Tr}_{K'\supseteq K}(\mu),\end{aligned}$$

   and thus $\sum_{P'|P}\mathrm{Cotr}_{L'\supseteq L}(\omega)_{P'}(f) = \omega_P(f)$.

□

Proof of Theorem 4.1: To start with, the inclusion "$\supseteq$" is proven. Take

$$\mathbf{a} = \Big(\mathrm{Cotr}_{F_{k_1}\supseteq F}(\omega)_{P_1^*}(1),\ldots,\mathrm{Cotr}_{F_{k_n}\supseteq F}(\omega)_{P_n^*}(1)\Big)\in C_\Omega(D,G)$$

and $\mathbf{c} = \big(f(P_1),\ldots,f(P_n)\big) = \big(f(P_1^*),\ldots,f(P_n^*)\big)\in C_{\mathcal{L}}(D,G)$. Then

$$\begin{aligned}\langle\mathbf{a},\mathbf{c}\rangle &= \sum_{i=1}^n \mathrm{Tr}_{\mathbb{F}_{q^{k_i}}\supseteq\mathbb{F}_q}\left(\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)_{P_i^*}(1)\cdot f(P_i)\right)\\ &= \sum_{i=1}^n \mathrm{Tr}_{\mathbb{F}_{q^{k_i}}\supseteq\mathbb{F}_q}\left(\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)_{P_i^*}(1)\cdot f(P_i^*)\right)\\ &= \sum_{i=1}^n\sum_{\sigma\in\mathrm{Gal}(F_{k_i}\supseteq F)}\sigma\left(\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)_{P_i^*}(1)\cdot f(P_i^*)\right)\\ &= \sum_{i=1}^n\sum_{\sigma\in\mathrm{Gal}(F_{k_i}\supseteq F)}\sigma\left(\mathrm{Cotr}_{F_{k_i}\supseteq F}(\omega)_{P_i^*}(1)\right)\cdot\sigma\left(f(P_i^*)\right)\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i=1}^{n} \sum_{\sigma \in \mathrm{Gal}(F_{k_i} \supseteq F)} \mathrm{Cotr}_{F_{k_i} \supseteq F}(\omega)_{\sigma(P_i^*)}(1) \cdot \big(f + \sigma(P_i^*)\big) && \text{(Lemma 4.2.2)} \\
&= \sum_{i=1}^{n} \sum_{Q_i^* | P_i} \mathrm{Cotr}_{F_{k_i} \supseteq F}(\omega)_{Q_i^*}(1) \cdot f(Q_i^*) && \text{(Lemma 4.2.1)} \\
&= \sum_{i=1}^{n} \sum_{Q_i' | P_i} \mathrm{Cotr}_{F' \supseteq F_{k_i}} \big(\mathrm{Cotr}_{F_{k_i} \supseteq F}(\omega)\big)_{Q_i'}(1) \cdot f(Q_i') && \text{(Lemma 4.2.3)} \\
&= \sum_{i=1}^{n} \sum_{Q_i' | P_i} \mathrm{Cotr}_{F' \supseteq F}(\omega)_{Q_i'}(1) \cdot f(Q_i') \\
&= 0
\end{aligned}
$$

since $C_{\mathcal{L}}(D', G')^{\perp} = C_{\Omega}(D', G')$, $\mathrm{Cotr}_{F' \supseteq F}(\omega) \in \Omega_{F'}(G' - D')$ when $\omega \in \Omega_F(G - D)$ and thus $(\mathrm{Cotr}_{F' \supseteq F}(\omega)_{Q_i'}(1) \mid Q_i' | P_i,\ 1 \leq i \leq n) \in C_{\Omega}(D', G')$, $(f(Q_i') \mid Q_i' | P_i,\ 1 \leq i \leq n) \in C_{\mathcal{L}}(D', G')$.

To prove equality, it is shown that the dimensions of the two spaces are equal:

$$
\begin{aligned}
\dim C_{\Omega}(D, G) &= i(G - D) - i(G) \\
&= \dim \mathcal{L}(G - D) - \deg(G - D) + g - 1 \\
&\qquad - \big(\dim \mathcal{L}(G) - \deg G + g - 1\big) \\
&= \deg D + \dim \mathcal{L}(G - D) - \dim \mathcal{L}(G) \\
&= N - \dim C_{\mathcal{L}}(D, G) \\
&= \dim C_{\mathcal{L}}(D, G)^{\perp}
\end{aligned}
$$

□

The proof makes apparent that codewords of $C_{\mathcal{L}}(D, G)$ and $C_{\Omega}(D, G)$ can be regarded as codewords of $C_{\mathcal{L}}(D', G')$ and $C_{\Omega}(D', G')$. If $\mathbf{c} = (c_1, \ldots, c_n)$ is a codeword of $C_{\mathcal{L}}(D, G)$, then $\mathbf{c}' = \big(\sigma(c_i) \mid \sigma \in \mathrm{Gal}(\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q),\ 1 \leq i \leq n\big)$ is a codeword of $C_{\mathcal{L}}(D', G')$ where there is a fixed order on the $\sigma \in \mathrm{Gal}(\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q)$ corresponding to the chosen fixed order on the places $Q_i' | P_i$. The same holds for codewords of $C_{\Omega}(D, G)$.

In the remainder of this section, it is shown that, as to be expected, codes of the form $C_{\Omega}(D, G)$ are generalized geometric Goppa codes.

Proposition 4.3 *If there exists a Weil differential $\omega \in \Omega_F$ such that $v_{P_i}(\omega) = -1$ and $\mathrm{Cotr}_{F_{k_i} \supseteq F}(\omega)_{P_i^*}(1) = 1$ for $i = 1, \ldots, n$, then*

$$C_{\mathcal{L}}(D, G)^{\perp} = C_{\Omega}(D, G) = C_{\mathcal{L}}\big(D, D - G + (\omega)\big).$$

Proof: Note that $v_{P_i}(\omega) = -1$ for $i = 1, \ldots, n$ implies that $\mathrm{supp}\,\big(D - G + (\omega)\big) \cap \mathrm{supp}\, D = \emptyset$. Hence $C_{\mathcal{L}}\big(D, D - G + (\omega)\big)$ is defined. By (Stichtenoth 1993, I.5.1), there exists an isomorphism $\mathcal{L}\big(D - G + (\omega)\big) \to \Omega_F(G - D)$ defined

by $x \mapsto x\omega$. Note that

$$\begin{aligned}\mathrm{Cotr}_{F_{k_i} \supseteq F}(x\omega)_{P_i^*}(1) &= \big(x\mathrm{Cotr}_{F_{k_i} \supseteq F}(\omega)\big)_{P_i^*}(1) \\ &= \mathrm{Cotr}_{F_{k_i} \supseteq F}(\omega)_{P_i^*}(x) \\ &= x(P_i^*)\mathrm{Cotr}_{F_{k_i} \supseteq F}(\omega)_{P_i^*}(1) \\ &= x(P_i)\end{aligned}$$

for $i = 1, \ldots, n$. This implies $C_\Omega(D, G) = C_\mathcal{L}\big(D, D - G + (\omega)\big)$. □

Again a differential ω as required in Proposition 4.3 can be constructed using (Stichtenoth 1993, VII.1.2) if the existence of certain ingredients is ensured. Therefore the following lemma.

Lemma 4.3 *There exist $z, t \in F$ such that $v_{P_i}(z) = 0$, $z(P_i) = 1$ and $v_{P_i}(t) = 1$ for $i = 1, \ldots, n$.*

Proof : By the Approximation Theorem, there exists $z \in F$ such that $v_{P_i}(z - 1) > 0$ for $i = 1, \ldots, n$, which implies the desired properties of z. Similarly, by the Approximation Theorem, there exists $t \in F$ such that $v_{P_i}(t) = 1$ for $i = 1, \ldots, n$. □

Proposition 4.4 *Let $z, t \in F$ such that $v_{P_i}(z) = 0$, $z(P_i) = 1$ and $v_{P_i}(t) = 1$ for $i = 1, \ldots, n$. Then the Weil differential $\omega = z/t \cdot dt \in \Omega_F$ satisfies*

$$v_{P_i}(\omega) = -1 \quad \textit{and} \quad \mathrm{Cotr}_{F_{k_i} \supseteq F}(\omega)_{P_i^*}(1) = 1 \quad \textit{for} \quad i = 1, \ldots, n$$

and thus

$$C_\mathcal{L}(D, G)^\perp = C_\Omega(D, G) = C_\mathcal{L}\big(D, D - G + (z) + (dt) - (t)\big).$$

The central point of the above proposition is the fact that the Weil differential $\omega' = z/t \cdot dt \in \Omega_{F'}$ is equal to $\mathrm{Cotr}_{F' \supseteq F}(\omega)$. This is what is proven in the following lemma. Note that depending on the context, dt denotes the exact differential with respect to t of either $F \supseteq \mathbb{F}_q$ or $F' \supseteq \mathbb{F}_{q^\ell}$.

Lemma 4.4 *Let $z, t \in F$ such that $v_{P_i}(z) = 0$, $z(P_i) = 1$ and $v_{P_i}(t) = 1$ for $i = 1, \ldots, n$. Then the Weil differentials $\omega = z/t \cdot dt \in \Omega_F$ and $\omega' = z/t \cdot dt \in \Omega_{F'}$ satisfy*

$$\omega' = \mathrm{Cotr}_{F' \supseteq F}(\omega).$$

Proof : Note that t is a separating element of both $F \supseteq \mathbb{F}_q$ and $F' \supseteq \mathbb{F}_{q^\ell}$ by (Stichtenoth 1993, III.9.2 (a)), and therefore dt is non-trivial in either algebraic function field.

Let η and η' be the unique Weil differentials (Stichtenoth 1993, I.7.4) of the rational function fields $\mathbb{F}_q(t) \supseteq \mathbb{F}_q$ and $\mathbb{F}_{q^\ell}(t) \supseteq \mathbb{F}_{q^\ell}$ with the properties

$$\begin{aligned}(\eta) &= -2P_\infty, & (\eta') &= -2P'_\infty, \\ \eta_{P_\infty}(t^{-1}) &= -1, & \eta'_{P'_\infty}(t^{-1}) &= -1,\end{aligned}$$

where P_∞ and P'_∞ denote the infinite places of the rational function fields $\mathbb{F}_q(t) \supseteq \mathbb{F}_q$ and $\mathbb{F}_{q^\ell}(t) \supseteq \mathbb{F}_{q^\ell}$ respectively. It is known that

$$\omega = \frac{z}{t} \cdot dt = \frac{z}{t} \cdot \mathrm{Cotr}_{F \supseteq \mathbb{F}_q(t)}(\eta) \quad \text{and} \quad \omega' = \frac{z}{t} \cdot dt = \frac{z}{t} \cdot \mathrm{Cotr}_{F' \supseteq \mathbb{F}_{q^\ell}(t)}(\eta').$$

The function field $\mathbb{F}_{q^\ell}(t) \supseteq \mathbb{F}_{q^\ell}$ is a constant field extension of $\mathbb{F}_q(t) \supseteq \mathbb{F}_q$ and therefore $\mathrm{Cotr}_{\mathbb{F}_{q^\ell}(t) \supseteq \mathbb{F}_q(t)}(\eta)$ has the same two properties as η' by Lemma 4.2.3. But η' is unique with these properties and thus $\eta' = \mathrm{Cotr}_{\mathbb{F}_{q^\ell}(t) \supseteq \mathbb{F}_q(t)}(\eta)$.

By calculation it is established that

$$\begin{aligned}
\omega' &= \frac{z}{t} \cdot \mathrm{Cotr}_{F' \supseteq \mathbb{F}_{q^\ell}(t)}(\eta') = \frac{z}{t} \cdot \mathrm{Cotr}_{F' \supseteq \mathbb{F}_{q^\ell}(t)}\big(\mathrm{Cotr}_{\mathbb{F}_{q^\ell}(t) \supseteq \mathbb{F}_q(t)}(\eta)\big) \\
&= \frac{z}{t} \cdot \mathrm{Cotr}_{F' \supseteq \mathbb{F}_q(t)}(\eta) = \mathrm{Cotr}_{F' \supseteq F}\big(\frac{z}{t} \cdot \mathrm{Cotr}_{F \supseteq \mathbb{F}_q(t)}(\eta)\big) \\
&= \mathrm{Cotr}_{F' \supseteq F}(\omega).
\end{aligned}$$

□

Proof of Proposition 4.4: Note that $z, t \in F \subseteq F'$ with properties $v_{P_i}(z) = 0$, $z(P_i) = 1$ and $v_{P_i}(t) = 1$ also satisfy $v_{P'_i}(z) = 0$, $z(P'_i) = 1$ and $v_{P'_i}(t) = 1$ for $i = 1, \ldots, n$. All P'_i are of degree 1. Therefore, the Weil differential $\omega' = z/t \cdot dt \in \Omega_{F'}$ satisfies $v_{P'_i}(\omega') = -1$ and $\mathrm{res}_{P'_i}(\omega') = 1$ by (Stichtenoth 1993, VII.1.2). But then

$$\begin{aligned}
-1 = v_{P'_i}(\omega') &= v_{P'_i}\big(\mathrm{Cotr}_{F' \supseteq F}(\omega)\big) = v_{P_i}(\omega) \qquad \text{and} \\
1 = \mathrm{res}_{P'_i}(\omega') &= \omega'_{P'_i}(1) = \mathrm{Cotr}_{F' \supseteq F}(\omega)_{P'_i}(1) = \mathrm{Cotr}_{F_{k_i} \supseteq F}(\omega)_{P_i^*}(1).
\end{aligned}$$

Thus the first claim of the proposition is proven. The second claim follows immediately from Proposition 4.3. □

4.5 The Minimum Distance Revisited

In some cases it is possible to improve the bound on the minimal distance of the generalized geometric Goppa code, due to the correspondence of codewords in the generalized geometric Goppa code C associated with D and G (either $C_{\mathcal{L}}(D, G)$ or $C_\Omega(D, G)$) to certain codewords of the conventional one C' associated with D' and G'. Let d^* be the designed minimum distance of C'. This means $d^* = N - \deg G$ if $C = C_{\mathcal{L}}(D, G)$ is considered and $d^* = \deg G - (2 - g)$ if $C = C_\Omega(D, G)$ is considered.

From now on, let $M = \max\{k_i \mid 1 \le i \le n\}$. Further, let r_j, $1 \le j \le M$ be the number of places P_i of degree $k_i = j$. Define the integers $d_{0j} = \min\big\{r_j, \lfloor (d^* - \sum_{s=j+1}^{M} s d_{0s})/j \rfloor\big\}$, $1 \le j \le M$. If $\sum_{j=1}^{M} j d_{0j} < d^*$, then find s as large as possible such that $d_s = d_{0s} + 1 \le r_j$ and define further $d_j = d_{0j}$ for $s \ne j$. Otherwise $d_j = d_{0j}$ for all j. Therefore $\sum_{j=1}^{M} j d_j \ge d^*$.

Proposition 4.5 *The code C has minimum distance*

$$d \ge \sum_{j=1}^{M} d_j \ge \left\lceil \frac{d^*}{M} \right\rceil.$$

Proof : Let $\mathbf{c} = (c_1, \ldots, c_n) \in C$. The aim is to estimate how small the number $\mathrm{w}(\mathbf{c})$ can get. The corresponding word $c' = \big(\sigma(c_i) \mid \sigma \in \mathrm{Gal}(\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q),\ 1 \leq i \leq n\big) \in C'$ is of weight $\geq d^*$.

Consider $\mathbf{a} = (a_1, \ldots, a_n) \in \prod_{i=1}^{n} \mathbb{F}_{q^{k_i}}$ and correspondingly $\mathbf{a}' = \big(\sigma(a_i) \mid \sigma \in \mathrm{Gal}(\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q),\ 1 \leq i \leq n\big) \in \mathbb{F}_{q^\ell}^N$ such that $\mathrm{w}(\mathbf{a}') \geq d^*$ is as small as possible while the non-zero positions of $\mathbf{a}$ occupy as many positions corresponding to places of high degree as possible. Note that if $a_i \neq 0$, then $\sigma(a_i) \neq 0$ for $\sigma \in \mathrm{Gal}(\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q)$ and if one entry of $\mathbf{a}'$ corresponding to a place above P_i is non-zero, it means that all entries corresponding to places above P_i are non-zero. Therefore $\mathbf{a}$ is of lowest possible weight with the restriction $\mathrm{w}(\mathbf{a}') \geq d^*$. It satisfies these criteria if $\mathrm{w}(\mathbf{a}) = \sum_{j=1}^{M} d_j$ and thus $\mathrm{w}(\mathbf{a}') = \sum_{j=1}^{M} j d_j$.

It can be concluded that $\mathrm{w}(\mathbf{c}) \geq \mathrm{w}(\mathbf{a}) = \sum_{j=1}^{M} d_j \geq \lceil d^*/M \rceil$. □

Consider a generalized geometric Goppa code defined on the function field $F \supseteq \mathbb{F}_q$, and let n, N and M be fixed integers. Whether the bound obtained in Proposition 4.5 is better than the one from Proposition 4.1 or Proposition 4.2 depends on $\deg G$.

Let $d^*_{\mathcal{L}}$ be the best lower bound on the minimum distance $d_{\mathcal{L}}$ of $C_{\mathcal{L}}(D, G)$. Then the bound on the code $C_{\mathcal{L}}(D, G, C_1, \ldots, C_n)$ can be improved slightly.

Proposition 4.6 *The minimum distance d' of the code $C_{\mathcal{L}}(D, G, C_1, \ldots, C_n)$ over $\mathbb{F}_q$ satisfies*

$$d' \geq \sum_{i=1}^{n} d_i - \deg G - \max\left\{ \sum_{i \in I} (d_i - k_i) \mid I \subseteq \{1, \ldots, n\}, |I| \leq n - d^*_{\mathcal{L}} \right\}.$$

Proof : The proof is analogous to the one of the relevant part of (Xing et al.1999, Theorem 3.2) with the additional information that the number of places a function of $\mathcal{L}(G)$ vanishes on cannot exceed $d^*_{\mathcal{L}}$. □

Certainly, this new bound can be an improvement only if there exists $i \in \{1, \ldots, n\}$ such that $d_i > k_i$. Further, a corresponding result can be stated immediately for the code $C_\Omega(D, G, C_1, \ldots, C_n)$ due to the relation between $C_{\mathcal{L}}(D, G)$ and $C_\Omega(D, G)$.

4.6 Decoding

As seen in the previous section codewords of $C_{\mathcal{L}}(D, G)$ and $C_\Omega(D, G)$ are closely related to certain codewords of $C_{\mathcal{L}}(D', G')$ and $C_\Omega(D', G')$. This property can be made use of when decoding generalized geometric Goppa codes. Suppose $\mathbf{a} = (a_1, \ldots, a_n) = \mathbf{c} + \mathbf{e} \in \prod_{i=1}^{n} \mathbb{F}_{q^{k_i}}$, $\mathbf{c} = (c_1, \ldots, c_n) \in C_\Omega(D, G)$ is received to be decoded to $\mathbf{c}$. The idea of the following is that an equivalent task is to decode $\mathbf{a}' = \big(\sigma(a_i) \mid \sigma \in \mathrm{Gal}(\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q),\ 1 \leq i \leq n\big)$ to $\mathbf{c}' = \big(\sigma(c_i) \mid \sigma \in \mathrm{Gal}(\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q),\ 1 \leq i \leq n\big) \in C_\Omega(D', G')$. The method presented is closely related to (Stichtenoth 1993, VII.5) and its advantage is that the computations

can be done over the field $\mathbb{F}_q$ rather than $\mathbb{F}_{q^\ell}$ as opposed to decoding $\mathbf{a}'$ directly to an element of $C_\Omega(D', G')$.

Let us consider the code $C_\Omega(D, G)$. Its dual is $C_\mathcal{L}(D, G)$, so the elements of $C_\mathcal{L}(D, G)$ can be considered as parity checks. Therefore the following definition:

Definition 4.7 *For* $\mathbf{a} = (a_1, \ldots, a_n) \in \prod_{i=1}^n \mathbb{F}_{q^{k_i}}$ *and* $f \in \mathcal{L}(G)$ *define the* syndrome

$$\langle \mathbf{a}, f \rangle = \sum_{i=1}^{n} \mathrm{Tr}_{\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q} \big(a_i f(P_i)\big).$$

The map $\langle \cdot, \cdot \rangle$ is bilinear due to the properties of the trace function. Note $C_\Omega(D, G) = \{\mathbf{c} \in \prod_{i=1}^n \mathbb{F}_{q^{k_i}} \mid \langle \mathbf{c}, f \rangle = 0 \text{ for all } f \in \mathcal{L}(G)\}$.

Let $\epsilon \geq 0$ be an integer and G_0 a divisor of $F \supseteq \mathbb{F}_q$ satisfying the following conditions:

$$\begin{cases} \operatorname{supp} G_0 \cap \operatorname{supp} D = \emptyset \\ \deg G_0 < \deg G - (2g-2) - M\epsilon \\ \dim \mathcal{L}(G_0) > M\epsilon \end{cases} \tag{4.1}$$

Lemma 4.5 *Let* $d^* = \deg G' - (2g-2) = \deg G - (2g-2)$ *be the designed minimum distance of* $C_\Omega(D', G')$.

1. *If* ϵ *and* G_0 *satisfy (4.1) then* $M\epsilon \leq (d^* - 1)/2$.
2. *If* $0 \leq M\epsilon \leq (d^* - 1 - g)/2$, *then there exists a divisor* G_0 *such that (4.1) is satisfied.*

Proof :

1. On the one hand, $M\epsilon \leq \dim \mathcal{L}(G_0) - 1 \leq \deg G_0$, on the other $M\epsilon \leq \deg G - (2g-2) - 1 - \deg G_0 = d^* - 1 - \deg G_0$. Adding the two inequalities gives $M\epsilon \leq (d^* - 1)/2$.
2. Choose G_0 such that $\deg G_0 = g + M\epsilon$ and $\operatorname{supp} G_0 \cap \operatorname{supp} D = \emptyset$. Then $\dim \mathcal{L}(G_0) \geq \deg G_0 - g + 1 = M\epsilon + 1 > M\epsilon$. Further, $\deg G - (2g-2) - M\epsilon - \deg G_0 = d^* - 2M\epsilon - g > 0$ by the assumption.

□

Assume in the following that (4.1) is satisfied. The next proposition will show how a function $g \in \mathcal{L}(G_0)$ can be created that will give an indication of the error positions since $g(P_i) = 0$ for all error positions i.

Proposition 4.7 *Let* $\mathbf{a} = \mathbf{c} + \mathbf{e} \in \prod_{i=1}^n \mathbb{F}_{q^{k_i}}$, $\mathbf{e} = (e_1, \ldots, e_n)$ *with* $\mathbf{c} \in C_\Omega(D, G)$, $\mathrm{w}(\mathbf{e}) \leq \epsilon$ *and* $\{g_1, \ldots, g_l\}$ *be a basis of* $\mathcal{L}(G_0)$, $\{h_1, \ldots, h_m\}$ *a basis of* $\mathcal{L}(G - G_0)$. *The homogeneous system of linear equations with unknowns* $\mu_1, \ldots, \mu_l$

$$\sum_{s=1}^{l} \langle \mathbf{a}, g_s h_u \rangle \, \mu_s = 0, \qquad u = 1, \ldots, m \tag{4.2}$$

has a non-trivial solution in $\mathbb{F}_q^l$. *If* $(\mu_1, \ldots, \mu_l)$ *is such a solution, set* $g = \sum_{s=1}^l \mu_s g_s \in \mathcal{L}(G_0)$. *Then* $g(P_i) = 0$ *for all* i *such that* $e_i \neq 0$.

Proof : Note that $g_\mu h_u \in \mathcal{L}(G)$, so $\langle \mathbf{a}, g_s h_u \rangle$ is well defined.

Let $I = \{i \mid 1 \leq i \leq n,\, e_i \neq 0\}$. Since $|I| \leq \epsilon$ and $\dim \mathcal{L}(G_0) > M\epsilon$, $\dim \mathcal{L}(G_0 - \sum_{i\in I} P_i) > 0$ follows. Choose $0 \neq g \in \mathcal{L}(G_0 - \sum_{i\in I} P_i)$ and write $g = \sum_{s=1}^{l} \mu_s g_s$ with $\mu_s \in \mathbb{F}_q$. Therefore

$$\begin{aligned}
\sum_{s=1}^{l} \langle \mathbf{a}, g_s h_u \rangle \mu_s &= \sum_{s=1}^{l} \mu_s \sum_{i=1}^{n} \mathrm{Tr}_{\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q} \big(a_i g_s(P_i) h_u(P_i)\big) \\
&= \sum_{i=1}^{n} \mathrm{Tr}_{\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q} \Big(a_i \big(\sum_{s=1}^{l} \mu_s g_s(P_i)\big) h_u(P_i)\Big) \\
&= \sum_{i=1}^{n} \mathrm{Tr}_{\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q} \big(a_i g(P_i) h_u(P_i)\big) \\
&= \langle \mathbf{a}, g h_u \rangle = \langle \mathbf{c} + \mathbf{e}, g h_u \rangle = \langle \mathbf{e}, g h_u \rangle \\
&= \sum_{i=1}^{n} \mathrm{Tr}_{\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q} \big(e_i g(P_i) h_u(P_i)\big) \\
&= 0.
\end{aligned}$$

Therefore $(\mu_1, \ldots, \mu_l)$ is a non-trivial solution of (4.2).

Let $(\mu_1, \ldots, \mu_l)$ be an arbitrary solution and $g = \sum_{s=1}^{l} \mu_s g_s$. Suppose there exists $i_0 \in I$ such that $g(P_{i_0}) \neq 0$. Observe

$$\deg \big(G - G_0 - \sum_{i\in I} P_i\big) \geq \deg G - \deg G_0 - M\epsilon > 2g - 2.$$

This implies

$$\mathcal{L}\big(G - G_0 - \sum_{i\in I} P_i\big) \subsetneqq \mathcal{L}\big(G - G_i - \sum_{i\in I\setminus\{i_0\}} P_i\big)$$

and an element $h = \sum_{u=1}^{m} \nu_u h_u \in \mathcal{L}(G - G_0)$ can be found such that $h(P_{i_0}) \neq 0$ and $h(P_i) = 0$ for $i \in I \setminus \{i_0\}$. Now on the one hand

$$\langle \mathbf{a}, gh \rangle = \langle \mathbf{e}, gh \rangle = \sum_{i=1}^{n} e_i g(P_i) h(P_i) = e_{i_0} g(P_{i_0}) h(P_{i_0}) \neq 0,$$

but on the other

$$\langle \mathbf{a}, gh \rangle = \sum_{u=1}^{m} \nu_u \langle \mathbf{a}, g h_u \rangle = \sum_{u=1}^{m} \nu_u \sum_{s=1}^{l} \langle \mathbf{a}, g_s h_u \rangle \mu_s = 0.$$

A contradiction is obtained and therefore $g(P_i) = 0$ for all i such that $e_i \neq 0$. □

Proposition 4.8 *Again, let* $\mathbf{a} = \mathbf{c} + \mathbf{e} \in \prod_{i=1}^{n} \mathbb{F}_{q^{k_i}}$, $\mathbf{e} = (e_1, \ldots, e_n)$ *with* $\mathbf{c} \in C_\Omega(D, G)$, $\mathrm{w}(\mathbf{e}) \leq \epsilon$ *and* g *as in Proposition 4.7. Let further* $I_g = \{i \mid 1 \leq i \leq n$ *and* $g(P_i) = 0\}$, $\{f_1, \ldots, f_k\}$ *be a basis of* $\mathcal{L}(G)$ *and* $\big\{b_1^{(k_i)}, \ldots, b_{k_i}^{(k_i)}\big\}$

denote a fixed vector space basis of $\mathbb{F}_{q^{k_i}}$ *over* $\mathbb{F}_q$*. The system of linear equations with unknowns* λ_{ij}, $i \in I_g$

$$\sum_{i \in I_g} \sum_{j=1}^{k_i} \lambda_{ij} \mathrm{Tr}_{\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q} \left(f_r(P_i) b_j^{(k_i)} \right) = \langle \mathbf{a}, f_r \rangle \,, \qquad r = 1, \ldots, k \tag{4.3}$$

has a unique solution such that $\sum_{j=1}^{k_i} \lambda_{ij} b_j^{(k_i)} = e_i$, $i \in I_g$.

Proof: Write $e_i = \sum_{j=1}^{k_i} e_{ij} b_j^{(k_i)}$. Since $f_r \in \mathcal{L}(G)$,

$$\begin{aligned} \langle \mathbf{a}, f_r \rangle &= \langle \mathbf{c} + \mathbf{e}, f_r \rangle = \langle \mathbf{e}, f_r \rangle \\ &= \sum_{i \in I_g} \mathrm{Tr}_{\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q} \left(f_r(P_i) e_i \right) = \sum_{i \in I_g} \sum_{j=1}^{k_i} e_{ij} \mathrm{Tr}_{\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q} \left(f_r(P_i) b_j^{(k_i)} \right). \end{aligned}$$

Thus $(e_{ij} \in \mathbb{F}_q \mid 1 \le j \le k_i,\ i \in I_g)$ is a solution of (4.3).

Suppose $(\lambda_{ij} \in \mathbb{F}_q \mid 1 \le j \le k_i,\ i \in I_g)$ is another solution. Set $\lambda_i = \sum_{j=1}^{k_i} \lambda_{ij} b_j^{(k_i)}$, $i \in I_g$ and $\lambda_i = 0$ otherwise. Then $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n) \in \prod_{i=1}^n \mathbb{F}_{q^{k_i}}$. The above calculation makes apparent that $\langle \boldsymbol{\lambda}, f_r \rangle = \langle \mathbf{e}, f_r \rangle$ for $r = 1, \ldots, k$. Since $\{f_1, \ldots, f_k\}$ is a basis of $\mathcal{L}(G)$, this implies that $\boldsymbol{\lambda} - \mathbf{e} \in C_\Omega(D, G)$.

Let $\boldsymbol{\lambda}' = (\sigma(\lambda_i) \mid \sigma \in \mathrm{Gal}(\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q),\ 1 \le i \le n) \in \mathbb{F}_{q^\ell}^N$ and correspondingly $\mathbf{e}' = (\sigma(e_i) \mid \sigma \in \mathrm{Gal}(\mathbb{F}_{q^{k_i}} \supseteq \mathbb{F}_q),\ 1 \le i \le n) \in \mathbb{F}_{q^\ell}^N$. Also $\boldsymbol{\lambda}' - \mathbf{e}' \in C_\Omega(D', G')$. Further $\mathrm{w}(\boldsymbol{\lambda} - \mathbf{e}) \le |I_g|$ and $\mathrm{w}(\boldsymbol{\lambda}' - \mathbf{e}') \le \sum_{i \in I_g} k_i$. Since $0 \ne g \in \mathcal{L}\big(G_0 - \sum_{i \in I_g} P_i\big)$ implies $\deg G_0 \ge \sum_{i \in I_g} k_i$,

$$\mathrm{w}(\boldsymbol{\lambda}' - \mathbf{e}') \le \sum_{i \in I_g} k_i \le \deg G_0 < \deg G - (2g-2) - M\epsilon \le \deg G - (2g-2)$$

is obtained. Therefore $\boldsymbol{\lambda}' = \mathbf{e}'$, $\boldsymbol{\lambda} = \mathbf{e}$ and the solution of (4.3) is unique. □

The result of the propositions can be summarized in the following algorithm:

Input: $\mathbf{a} \in \prod_{i=1}^n \mathbb{F}_{q^{k_i}}$
Output: $\mathbf{c} \in C_\Omega(D, G)$ closest to $\mathbf{a}$

1: Find a non-trivial solution $(\mu_1, \ldots, \mu_l)$ to the homogeneous system of linear equations (4.2) and set $g = \sum_{s=1}^l \mu_s g_s$.
2: Determine $I_g = \{i \mid 1 \le i \le n \text{ and } g(P_i) = 0\}$.
3: If $(e_{ij} \in \mathbb{F}_q \mid 1 \le j \le k_i,\ i \in I_g)$ is a unique solution of (4.3), set $\mathbf{e} = (e_1, \ldots, e_n)$ with $e_i = \sum_{j=1}^{k_i} e_{ij} b_j^{(k_i)}$ for $i \in I_g$ and $e_i = 0$ otherwise.
4: Return $\mathbf{c} = \mathbf{a} - \mathbf{e}$.

Algorithm 4.1: Decoder for generalized geometric Goppa codes

Correctness of Algorithm 4.1 can be concluded directly from Lemma 4.5 and Propositions 4.7 and 4.8:

Corollary 4.1 *Consider the code $C_\Omega(D,G)$.*

1. *Provided ϵ and G_0 satisfy (4.1), Algorithm 4.1 decodes all error patterns of weight $\leq \epsilon$.*
2. *The divisor G_0 can be chosen such that Algorithm 4.1 can decode all error patterns $\mathbf{e}$ of weight*

$$\mathrm{w}(\mathbf{e}) \leq \big(\deg G - (2g-2) - 1 - g\big)/(2M). \tag{4.4}$$

Actually, more errors can be corrected than Corollary 4.1.2 suggests. It depends on how the errors are distributed across the error pattern. In the above, it would have been possible to introduce $\epsilon_{j_1}, \ldots, \epsilon_{j_w}$, $0 = j_0 < j_1 < \cdots < j_w = M$ instead of only ϵ, replacing all occurrences of $M\epsilon$ by $\sum_{r=1}^{w} j_r \epsilon_{j_r}$ and requiring that no more than ϵ_{j_r} errors have occurred on those positions that correspond to places of degree $j_{r-1}+1$ up to j_r. The above presentation is the special case with $w = 1$ and $j_1 = M$. The equivalent to Corollary 4.1 in this more general setup can be stated as follows:

Corollary 4.2 *Consider the code $C_\Omega(D,G)$.*

1. *Provided $\epsilon_{j_1}, \ldots, \epsilon_{j_w}$, $0 = j_0 < j_1 < \cdots < j_w = M$ and G_0 satisfy*

$$\begin{cases} \operatorname{supp} G_0 \cap \operatorname{supp} D = \emptyset, \\ \deg G_0 < \deg G - (2g-2) - \sum_{r=1}^{w} j_r \epsilon_{j_r}, \\ \dim \mathcal{L}(G_0) > \sum_{r=1}^{w} j_r \epsilon_{j_r}, \end{cases} \tag{4.1'}$$

   then Algorithm 4.1 can decode all error patterns of weight $\leq \sum_{r=1}^{w} \epsilon_{j_r}$ with $\leq \epsilon_{j_r}$ errors on positions corresponding to places P_i of degrees $j_{r-1} \leq k_i \leq j_r$.
2. *Let $0 = j_0 < j_1 < \cdots < j_w = M$. The divisor G_0 can be chosen such that Algorithm 4.1 can decode all error patterns of weight $\leq \sum_{r=1}^{w} \epsilon_{j_r}$ with $\leq \epsilon_{j_r}$ errors on positions corresponding to places P_i of degrees $j_{r-1} \leq k_i \leq j_r$ provided*

$$\sum_{r=1}^{w} j_r \epsilon_{j_r} \leq \big(\deg G - (2g-2) - 1 - g\big)/2.$$

The proof involves verifying corresponding versions of Lemma 4.5 and Propositions 4.7 and 4.8. In the Propositions, I and I_g need to be divided into w disjoint subsets corresponding to places of degrees $j_{r-1} \leq k_i \leq j_r$.

It is certainly possible to represent the generalized geometric Goppa code $C_\Omega(D,G)$ as a conventional k-dimensional code of length N over $\mathbb{F}_q$ by representing the elements of $\mathbb{F}_{q^{k_i}}$ in the ith position by the k_i coefficients corresponding to the basis $\{b_1^{(k_i)}, \ldots, b_{k_i}^{(k_i)}\}$ of the vector space $\mathbb{F}_{q^{k_i}}$ over $\mathbb{F}_q$. Denote this code by $C_\Omega(D,G)_{\mathbb{F}_q}$. This is the same as the concatenation of $C_\Omega(D,G)$ with the trivial $[k_i, k_i, 1]$ codes C_i over $\mathbb{F}_q$, resulting in $C_\Omega(D,G,C_1,\ldots,C_n)$. For $C_\Omega(D,G)_{\mathbb{F}_q}$, Corollary 4.1 implies that in general the same number of errors can be corrected as there can be in $C_\Omega(D,G)$. Corollary 4.2 implies that

depending on how the errors are distributed, as many errors can be decoded as in the geometric Goppa code $C_\Omega(D', G')$ over $\mathbb{F}_{q^\ell}$, although the lower bound on the minimum distance of $C_\Omega(D, G)_{\mathbb{F}_q}$ from Propositions 4.2 and 4.5 can be quite a bit smaller than the one of $C_\Omega(D', G')$. The reason is that any positive number of errors in the k_i positions corresponding to the place P_i all count as k_i errors in $C_\Omega(D', G')$ and as one error in the generalized geometric Goppa code $C_\Omega(D, G)$.

Again, consider the code

$$C_\Omega(D, G, C_1, \ldots, C_n) = C_{\mathcal{L}}(D, D - G + (\omega), C_1, \ldots, C_n)$$

for some Weil differential $\omega \in \Omega_L$. The ith position of the code $C_\Omega(D, G)$ is concatenated with an $\mathbb{F}_q$-code with parameters $[n_i, k_i, d_i]$. Suppose arbitrary error patterns of weight ϵ in the outer code can be corrected by Corollary 4.1. If decoding of the inner codewords results in $\leq \epsilon$ wrong inner words which the concatenated word consists of, then the errors of the concatenated word can be decoded. Corollary 4.2 implies that the concatenated word can be decoded if the number of remaining wrong inner words after inner decoding doesn't exceed the number of errors that can be corrected in the geometric Goppa code $C_\Omega(D', G')$, and if they are distributed across the concatenated word in the right way.

4.7 Example

Consider the Hermitian function field $H = \mathbb{F}_{r^2}(x, y) \supseteq \mathbb{F}_{r^2}$ defined by the equation $x^{r+1} = y^r + y$. This function field has genus $g = r(r-1)/2$ and the Zeta-function is

$$Z(t) = \frac{(1 + rt)^{2g}}{(1 - t)(1 - r^2 t)}.$$

Thus the numbers of places of degrees 1 to 3 are $B_1 = r^3 + 1$, $B_2 = 0$ and $B_3 = r^3(r^2 - 1)(r + 1)/3 > 0$.

The conventional Hermitian code is defined by

$$C_{\mathcal{L}}(D_1, sP_\infty) = \left\{ \big(f(P_1), \ldots, f(P_{B_1 - 1})\big) \mid f \in \mathcal{L}(sP_\infty) \right\} \subseteq \mathbb{F}_{r^2}^{B_1 - 1}$$

with $D_1 = \sum_{i=1}^{B_1 - 1} P_i$ where P_∞ is the infinite place and P_i, $1 \leq i \leq B_1 - 1$ are the distinct rational places of H not equal to P_∞. Its dual code is of interest since it gives us parity check equations, syndromes and a decoding algorithm. By (Stichtenoth 1993, VII.4.2), it is known that $C_{\mathcal{L}}(D_1, sP_\infty)^\perp = C_{\mathcal{L}}(D_1, (r^3 + r^2 - r - 2 - s)P_\infty)$.

Consider the generalized Hermitian code

$$\begin{aligned} &C_{\mathcal{L}}(D_2, sP_\infty) \\ &\quad = \left\{ \big(f(P_1), \ldots, f(P_{B_1 - 1}), f(P_{B_1}), \ldots, f(P_{B_1 + B_3 - 1})\big) \mid f \in \mathcal{L}(sP_\infty) \right\} \end{aligned}$$

with $D_2 = \sum_{i=1}^{B_1 - 1} P_i + \sum_{i=1}^{B_3} P_i$ where P_i, $B_1 \leq i \leq B_1 + B_3 - 1$ are the B_3 distinct places of degree 3 of H. Note that $C_{\mathcal{L}}(D_2, sP_\infty) \subseteq \mathbb{F}_{r^2}^{B_1 - 1} \times \mathbb{F}_{r^6}^{B_3}$ and

$\dim C_{\mathcal{L}}(D_2, sP_\infty) = \dim \mathcal{L}(sP_\infty) = s - r(r-1)/2 + 1$ if $r(r-1) - 2 < s < B_1 + 3B - 3 - 1 = r^6 + r^5 - r^4$.

Additionally, define the constant function field extension $H' = \mathbb{F}_{r^6} H \supseteq \mathbb{F}_{r^6}$, $D_2' = \mathrm{Con}_{H' \supseteq H}(D_2)$, $G' = \mathrm{Con}_{H' \supseteq H}(G)$ and P_∞' the only place above P_∞ by (Stichtenoth 1993, III.6.3). Then

$$C_{\mathcal{L}}(D_2', sP_\infty') \subseteq \mathbb{F}_{r^6}^{B_1 + 3B_3 - 1}$$

is a geometric Goppa code of length $N = B_1 - 1 + 3B_3 = r^6 + r^5 - r^4$, dimension $k_{\mathcal{L}} = \dim \mathcal{L}(sP_\infty') = \dim \mathcal{L}(sP_\infty) = s - r(r-1)/2 + 1$ if $r(r-1) - 2 < s < N = r^6 + r^5 - r^4$ which is assumed to hold from now on, and minimum distance $d_{\mathcal{L}} \geq r^6 + r^5 - r^4 - s$. By (Stichtenoth 1993, VII.1.2),

$$C_{\mathcal{L}}(D_2', sP_\infty')^\perp = C_\Omega(D_2', sP_\infty') = C_{\mathcal{L}}\big(D_2', D_2' - sP_\infty' + (z) + (dt) - (t)\big)$$

if $z, t \in H'$ are such that $v_{P_i}(z) = 0$, $z(P_i) = 1$ and t is a local parameter for all $1 \leq i \leq B_1 + B_3 - 1$. Obviously, $z = 1 \in H \subseteq H'$ can be chosen, since then $(z) = 0$ and $z(P_i) = 1$. Consider

$$t = \prod_{\alpha \in \mathbb{F}_{r^6}} (x - \alpha) = x^{r^6} - x \in H \subseteq H',$$

which is a local parameter for each P_i'. Then $dt = -dx$ (Stichtenoth 1993, IV.1.2) and by (Stichtenoth 1993, VI.4.1) $(dt) = (dx) = (r^2 - r - 2)P_\infty'$. Furthermore $(t) = D_2' - (B_1 + 3B_3 - 1)P_\infty' = D_2' - (r^6 + r^5 - r^4)P_\infty'$, which becomes apparent when t is regarded as an element of $\mathbb{F}_{r^6}(x)$. Thus

$$C_{\mathcal{L}}(D_2', sP_\infty')^\perp = C_\Omega(D_2', sP_\infty') = C_{\mathcal{L}}\big(D_2', (r^6 + r^5 - r^4 + r^2 - r - 2 - s)P_\infty'\big)$$

and $\dim C_{\mathcal{L}}(D_2', sP_\infty')^\perp = r^6 + r^5 - r^4 + r(r-1)/2 - 1 - s$ as to be expected.

Set $s^\perp = r^6 + r^5 - r^4 + r^2 - r - 2 - s$, then $r(r-1) - 2 < s^\perp < N = r^6 + r^5 - r^4$ as well and

$$C_{\mathcal{L}}(D_2, sP_\infty)^\perp = C_{\mathcal{L}}(D_2, s^\perp P_\infty) = C_{\mathcal{L}}\big(D_2, D_2 - sP_\infty + (z) + (dt) - (t)\big).$$

When it comes to decoding $C_\Omega(D_2, sP_\infty) = C_{\mathcal{L}}(D_2, s^\perp P_\infty)$, constructing the divisor G_0 under the conditions of Lemma 4.5 is straightforward. The choice $G_0 = \big(r(r-1)/2 + 3\epsilon\big)P_\infty$ will satisfy (4.1). According to Corollary 4.1.2, up to $\big(s - 3r(r-1)/2 + 1\big)/6$ errors can be corrected.

For a concrete example, consider $r = 2$. The Hermitian function field is of genus $g = 1$. Apart from the infinite place, it has 8 places of degree 1 and 24 places of degree 3. If $s = 70$ is chosen, then the dimension of $C_{\mathcal{L}}(D_2, 70P_\infty)$ is $k_{\mathcal{L}} = 70$ and $s^\perp = 10$. By Corollary 4.1.2, any error pattern with up to 11 errors can be corrected. Corollary 4.2.2 on the other hand implies that also error patterns with up to 4 errors in positions corresponding to places of degree 1 and up to 10 in positions corresponding to places of degree up to 3 can be corrected, and similarly up to 7 of degree 1 and 9 of degree up to 3, up to 8 of degree 1 (there are only 8) and 8 of degree up to 3.

4.8 Conclusions

The main contribution of this work is the extension of the concept of a code and its dual to $\mathbb{F}_q$-vector spaces over a mixed alphabet and the investigations into the dual of a generalized geometric Goppa code. Once this dual code is defined, it becomes apparent that both code and dual are related to each other like conventional geometric Goppa codes. The understanding that elements of the codes can be regarded as codewords of a conventional geometric Goppa code results in the byproduct of a possibly improved bound on the minimum distance of the code and thus of the concatenated code by Xing et al. (1999). Subsequently, a decoding algorithm for generalized geometric Goppa codes is derived. As these codes can be considered as the outer code of the construction of concatenated generalized geometric Goppa codes, decoding of these conventional codes over a finite field is made possible.

For further reading on generalized geometric Goppa codes: Another decoding algorithm with improved error correction capacity has been developed in (Heydtmann 2001). Further, a scheme for the concatenated codes that takes decoding information from the inner code into account when using the algorithm from (Heydtmann 2001) is presented in (Nielsen 2001).

References

Brouwer, A. E. (1998). Bounds on the Size of Linear Codes. In V. S. Pless and W. Huffman (Eds.), *Handbook of Coding Theory*, pp. 295–461. North Holland. Updated tables are available at `http://www.win.tue.nl/~aeb/voorlincod.html`.

Ding, C., H. Niederreiter, and C. Xing (2000). Some New Codes from Algebraic Curves. *IEEE Transactions on Information Theory 46*(7), 2638–2642.

Ericson, T. (1988). A Simple Analysis of the Blokh-Zyablov Decoding Algorithm. In T. Beth and M. Clausen (Eds.), *Applicable Algebra, Error-Correcting Codes, Combinatorics and Computer Algebra, 4th International Conference, Proceedings*, Volume 307 of *Lecture Notes in Computer Science*, pp. 43–57. Springer.

Farran, J. I. (2000). Decoding Algebraic Geometry Codes by a Key Equation. *Finite Fields and Their Applications 6*, 207–217.

Garcia, A. and H. Stichtenoth (1995). A Tower of Artin-Schreier Extensions of Function Fields Attaining the Drinfeld-Vladut Bound. *Inventiones mathematicae 121*, 211–222.

Goppa, V. D. (1981). Codes over Algebraic Curves. *Soviet Math.* Dokl. *24*(1), 170–172.

Heydtmann, A. E. (2001). Sudan-Decoding Generalized Geometric Goppa Codes. Submitted to *Finite Fields and Their Applications*.

MacWilliams, F. J. and N. J. A. Sloane (1977). *The Theory of Error-Correcting Codes*. North-Holland Mathematical Library.

Nielsen, R. R. (2001). Decoding Generalized Algebraic-Geometry Codes. manuscript.

Stichtenoth, H. (1993). *Algebraic Function Fields and Codes*. Springer.

Tsfasman, M. A., S. G. Vladut, and T. Zink (1982). Modular Curves, Shimura Curves and Goppa Codes better than the Varshamov-Gilbert Bound. *Mathematische Nachrichten 109*, 21–28.

Xing, C., H. Niederreiter, and K. Y. Lam (1999). A Generalization of Algebraic-Geometry Codes. *IEEE Transactions on Information Theory 45*(7), 2498–2501.

5. SUDAN-DECODING GENERALIZED GEOMETRIC GOPPA CODES

Agnes E. Heydtmann

Department of Mathematics, Building 303, Technical University of Denmark,
DK-2800 Kongens Lyngby, Denmark. Agnes.Heydtmann@mat.dtu.dk

Abstract

Generalized geometric Goppa codes are vector spaces of n-tuples with entries from different extension fields of a ground field. They are derived from evaluating functions similarly to conventional geometric Goppa codes, but allowing evaluation in places of arbitrary degree. A decoding scheme for these codes based on Sudan's improved algorithm is presented and its error-correcting capacity is analyzed. For the implementation of the algorithm it is necessary that so called increasing zero bases of certain spaces of functions are available. A method to obtain such bases is developed.

5.1 Introduction

Conventional geometric Goppa codes are constructed by evaluating functions of a given algebraic function field over a finite field in places of degree 1, so called rational places. It is the first construction of block codes that gives a sequence of codes asymptotically better than the Gilbert-Varshamov bound (Tsfasman et al. 1982; Garcia and Stichtenoth 1995).

Recently, a generalization of this construction has been put forth by Xing et al. (1999). Their construction uses places of arbitrary degree which functions are evaluated in. The result is a space of tuples with entries from different field extensions of the ground field. Each of these entries is subsequently concatenated with an inner codes. The inner codes may vary depending on the degree of the place evaluated. The authors give instances of codes from this construction whose parameters coincide with the best codes possible of that length and dimension, compare with (Brouwer 1998). Further, various examples of the construction that result in new improved codes given a fixed length and dimension are presented by Ding et al. (2000). This suggests that there might be another way to good codes than through the search for curves with many rational points.

This paper concentrates on what can be referred to as the outer code, namely the space of tuples with entries from different field extensions of the ground field. This outer code will be called the generalized geometric Goppa code in

the following and has been studied extensively in (Heydtmann 2000). A decoding algorithm for this space, i.e. a procedure to reobtain the original tuple when it has been disturbed by noise, will also result in a decoding algorithm of the concatenated codes by Xing et al. (1999) similarly to the decoding of conventional concatenated codes (MacWilliams and Sloane 1977; Ericson 1988) or with methods as in (Nielsen 2000; Nielsen 2001). In the following, it is demonstrated how Sudan's improved algorithm for decoding conventional geometric Goppa codes beyond half the minimum distance (Guruswami and Sudan 1999) can be extended to generalized geometric Goppa codes.

Section 5.2 introduces the necessary notation from the theory of algebraic function fields and presents the construction of generalized geometric Goppa codes as well as the concatenated version by Xing et al. (1999). The decoding algorithm is presented in Section 5.3 where also its correctness is proven and its error-correcting capability is analyzed. Section 5.4 supplements the decoding algorithm by a method for obtaining increasing zero bases of function spaces with respect to the places that are used for evaluation in the code. An example by Ding et al. (2000) is discussed in detail in Section 5.5 and concluding remarks can be found in Section 5.6.

5.2 Generalized Geometric Goppa Codes

The construction of generalized geometric Goppa codes requires some concepts from the theory of algebraic function fields. For an introduction to this subject, refer to (Stichtenoth 1993). Some basic notation is defined in the following.

Let $\mathbb{F}_q$ denote the finite field with q elements and let $F \supseteq \mathbb{F}_q$ be an algebraic function field of genus g.

The set of all places of F will be denoted by $\mathbb{P}_F$. The valuation ring of the place $P \in \mathbb{P}_F$ is $\mathcal{O}_P$ and v_P denotes the discrete valuation associated with P. Let $f \in F$. If $v_P(f) > 0$, then P is said to be a zero of multiplicity $v_P(f)$ of the function f and correspondingly if $v_P(f) < 0$ then P is a pole of multiplicity $-v_P(f)$ of f. Denote the residue class field of P by $F_P = \mathcal{O}_P/P$. The degree of P is $\deg P = [F_P : \mathbb{F}_q]$ and therefore $F_P \cong \mathbb{F}_{q^{\deg P}}$. Special places are the infinite places, denoted P_∞.

A divisor A of F is a formal sum of places of the form

$$A = \sum_{P \in \mathbb{P}_F} a_P P \quad \text{with } a_P \in \mathbb{Z} \text{ and almost all } a_P = 0.$$

The support of A is the set $\operatorname{supp} A = \{P \in \mathbb{P}_F \mid a_P \neq 0\}$ and the degree of the divisor A is defined by

$$\deg A = \sum_{P \in \mathbb{P}_F} a_P \deg P.$$

Let also $A' = \sum_{P \in \mathbb{P}_F} a'_P P$ be a divisor. A partial ordering on the group of divisors is given as follows:

$$A' \leq A \qquad \Longleftrightarrow \qquad a'_P \leq a_P \quad \forall\, P \in \mathbb{P}_F$$

For any $f \in F \setminus \{0\}$, the principal divisor of f is given by

$$(f) = \sum_{P \in \mathbb{P}_F} v_P(f)P = (f)_0 - (f)_\infty$$

where $(f)_0, (f)_\infty \geq 0$ called the zero divisor and the pole divisor of f respectively. If A is a divisor of F, then

$$\mathcal{L}(A) = \{f \in F \setminus \{0\} \mid (f) \geq -A\} \cup \{0\}$$

is a vector space over $\mathbb{F}_q$ and the dimension, denoted by $\dim A$, is bounded from below by the Riemann-Roch Theorem

$$\dim A \geq \deg A + 1 - g$$

with $\dim A = \deg A + 1 - g$ if $\deg A > 2g - 2$.

Let $D = P_1 + \cdots + P_n$ and G be divisors of F such that the P_i are distinct places of degree $\deg P_i = k_i$ and $\operatorname{supp} D \cap \operatorname{supp} G = \emptyset$. Denote the Cartesian product of the finite fields $\mathbb{F}_{q^{k_i}}$ by $\prod_{i=1}^n \mathbb{F}_{q^{k_i}}$, which is a $\deg D = \sum_{i=1}^n k_i$-dimensional $\mathbb{F}_q$-vector space. The *generalized geometric Goppa code* associated with the divisors D and G is

$$C_{\mathcal{L}}(D, G) = \{\big(f(P_1), \ldots, f(P_n)\big) \mid f \in \mathcal{L}(G)\},$$

considered as a subspace of $\prod_{i=1}^n \mathbb{F}_{q^{k_i}}$. Extending the definition of codes to subspaces of $\prod_{i=1}^n \mathbb{F}_{q^{k_i}}$ as above, the definitions for length, dimension, and Hamming distance and thus minimum distance can naturally be generalized: again, n is the length, the dimension of the code as a vector space over $\mathbb{F}_q$ is its dimension and the distance between two words is the number of differeing corresponding entries — compare with (Heydtmann 2000). Possibly, $k > n$. The code $C_{\mathcal{L}}(D, G)$ is an $[n, k, d]$ code with

$$k = \dim G - \dim(G - D), \qquad d \geq n - \deg G$$

and if $2g - 2 < \deg G < \deg D$, then $k = \deg G - g + 1$ by the Riemann-Roch Theorem. The original geometric Goppa code is the subclass of codes where $k_1 = \cdots = k_n = 1$.

A kind of concatenation of the above code is defined in (Xing et al. 1999). Let $C_1, \ldots, C_n$ be $[n_i, k_i, d_i]$ codes, $1 \leq i \leq n$. For each C_i there is an $\mathbb{F}_q$-linear isomorphism $\pi_i : \mathbb{F}_{q^{k_i}} \to C_i$, and the *concatenated generalized geometric Goppa code* is defined as follows:

$$C_{\mathcal{L}}\big(D, G, C_1, \ldots, C_n\big) = \big\{\big(\pi_1\big(f(P_1)\big), \ldots, \pi_n\big(f(P_n)\big)\big) \mid f \in \mathcal{L}(G)\big\}$$

If $\deg G < \deg D$, then by Xing et al. (1999, Theorem 3.2) $C_{\mathcal{L}}\big(D, G, C_1, \ldots, C_n\big)$ is an $\mathbb{F}_q$-linear code with parameters $[n', k', d']$ where $n' = \sum_{i=1}^r n_i$, $k' \geq \deg G + 1 - g$, equality holding when also $\deg G > 2g - 2$ and

$$d' \geq \sum_{i=1}^n d_i - \deg G - \max\left\{\sum_{i \in I}(d_i - k_i) \mid I \subseteq \{1, \ldots, n\}\right\}.$$

Note that $d' \geq \sum_{i=1}^{n} d_i - \deg G$ when $k_i \geq d_i$.

Some examples of the concatenated construction that result in better codes than previously known, i.e. codes with a greater minimum distance for a given length and dimension, are listed in (Ding et al. 2000). One such example will be discussed in greater depth in Section 5.5.

5.3 Decoding Generalized Geometric Goppa Codes

In this section the algorithm by Guruswami and Sudan (1999) will be extended to the class of generalized geometric Goppa codes as defined in the previous section. This will also make decoding of the code $C_{\mathcal{L}}\left(D, G, C_1, \ldots, C_n\right)$ possible. First some notation and concepts need introducing that are directly related to the decoding algorithm.

Let $Q \in F[Z]$ where Z is transcendental over F and write $Q = \sum_{j=0}^{\deg Q} q_j Z^j$ with $q_j \in F$. The evaluation of Q in $f \in F$ is defined by

$$Q(f) = \sum_{j=0}^{\deg Q} q_j f^j,$$

which is an element of the function field F. Let further $\mathbf{z} = (z_1, \ldots, z_n) \in \prod_{i=1}^{n} \mathbb{F}_{q^{k_i}}$ and if $q_j(P_i)$ is defined for all j, define the evaluation of Q in the pair (P_i, z_i) by

$$Q(P_i, z_i) = \sum_{j=0}^{\deg Q} q_j(P_i) z_i^j,$$

regarded as an element of $\mathbb{F}_{q^{k_i}}$ since $F_{P_i} \cong \mathbb{F}_{q^{k_i}}$. Let $\mathbb{N}$ denote the set of positive integers. The pair (P_i, z_i) is a *zero of multiplicity* $s_i \in \mathbb{N}$ of Q if for all $f \in F$ satisfying $f(P_i) = z_i$ the equality

$$v_{P_i}\big(Q(f)\big) = s_i$$

holds. Note that $Q(P_i, z_i) = Q(f)(P_i) = 0$ for any such function f.

The concept of what Høholdt and Nielsen (1999) refer to as an increasing zero basis of the space $\mathcal{L}(A)$ is needed. In this context it may however be with respect to a place P_i of arbitrary degree k_i. This involves extending (Guruswami and Sudan 1999, Lemma 21).

Proposition 5.1 *Let A be a divisor such that $\mathcal{L}(A)$ is non-trivial and $P_i \in \mathbb{P}_F$ with $k_i = \deg P_i$. Then there exists a basis*

$$\psi_1^{(i)}, \ldots, \psi_{\dim A}^{(i)}$$

of $\mathcal{L}(A)$ such that

$$v_{P_i}\big(\psi_1^{(i)}\big) \leq \cdots \leq v_{P_i}\big(\psi_{\dim A}^{(i)}\big)$$

with at most k_i elements of equal valuation.

A basis $\psi_1^{(i)}, \ldots, \psi_{\dim A}^{(i)}$ as in the proposition is called an *increasing zero basis* of $\mathcal{L}(A)$ with respect to P_i. The proof shows a method of obtaining such a basis from an arbitrary basis of $\mathcal{L}(A)$, provided that it is possible to find the unique representation of an element of F with respect to a local parameter of P_i. Section 5.4 will give such an algorithm for some function fields F.

Proof: Let $\phi_1, \ldots, \phi_{\dim A}$ form a basis of $\mathcal{L}(A)$ ordered such that

$$v_{P_i}(\phi_1) \leq \cdots \leq v_{P_i}(\phi_{\dim A}).$$

Let $\phi_1, \ldots, \phi_\iota$ be all the basis elements of smallest valuation $v_{P_i}(\phi_1)$. Given a local parameter $t_i \in P_i$ each of these has a unique representation $\phi_\alpha = u_\alpha t_i^{v_{P_i}(\phi_1)}$, $1 \leq \alpha \leq \iota$ for some a unit u_α in $\mathcal{O}_{P_i}$, i.e. $v_{P_i}(u_\alpha) = 0$ and $v_{P_i}(t_i) = 1$. Without loss of generality $u_1(P_i), \ldots, u_{\kappa_i}(P_i) \in \mathbb{F}_{q^{k_i}}$, $\kappa_i \leq \iota, k_i$ are linearly independent over $\mathbb{F}_q$ and $u_{\kappa_i+1}(P_i), \ldots, u_\iota(P_i)$ are linearly dependent of these. In other words there exist $\mu_{\alpha\beta} \in \mathbb{F}_q$ such that

$$u_\beta(P_i) = \sum_{\alpha=1}^{\kappa_i} \mu_{\alpha\beta} u_\alpha(P_i), \quad \beta = \kappa_i + 1, \ldots, \iota.$$

Note that $u_\beta - \sum_{\alpha=1}^{\kappa_i} \mu_{\alpha\beta} u_\alpha \neq 0$ due to the linear independence of the original basis elements and $v_{P_i}\left(u_\beta - \sum_{\alpha=1}^{\kappa_i} \mu_{\alpha\beta} u_\alpha\right) > 0$. Set

$$\begin{aligned} &\psi_1^{(i)} = \phi_1, \ldots, \psi_{\kappa_i}^{(i)} = \phi_{\kappa_i}, \\ &\phi'_\beta = \phi_\beta - \sum_{\alpha=1}^{\kappa_i} \mu_{\alpha\beta} \phi_\beta, \quad \beta = \kappa_i + 1, \ldots, \iota \end{aligned}$$

and observe that $v_{P_i}(\phi'_\beta) > v_{P_i}(\phi_1)$. Repeat this procedure with the linearly independent functions $\phi'_{\kappa_i+1}, \ldots, \phi'_\iota, \phi_{\iota+1}, \ldots, \phi_{\dim A}$. Eventually, a basis as desired is obtained. □

Suppose the word $\mathbf{z} = (z_1, \ldots, z_n) \in \prod_{i=1}^n \mathbb{F}_{q^{k_i}}$ is received and it should to be decoded to a codeword in $C_\mathcal{L}(D, G)$. Said informally, the idea of Sudan's algorithm is now to find a non-zero polynomial $Q \in F[Z]$ such that the pairs (P_i, z_i) are zeroes of Q of at least a given multiplicity s_i and the degree of Q as well as its coefficients are in a certain sense big enough to accommodate up to a fixed number of errors. If $f(P_i) = z_i$ for sufficiently many i, then it can be concluded that $Q(f)$ has too many zeroes for our choice of degree and coefficients of Q and therefore $Q(f) = 0$. This implies that $Z - f$ is a factor of Q and the functions corresponding to codewords closest to $\mathbf{z}$ are found by identifying linear factors of Q.

Algorithm 5.1 states the generalization of Sudan's algorithm precisely. By $\mathbb{N}_0$ in the input of the algorithm, the set of non-negative integers is denoted. First some remarks about the choices made in Step 1 of the algorithm: The important variables defined are λ, σ, s_i, H and d_Q whereas r, a, b, c, d'_Q and r' are introduced only to simplify expressions. Calculation shows that $b^2 - 4ac > 0$ and ϵ ought to be picked such that $a \neq 0$; compare with Theorem 5.1. Note

Input: n, P_i, k_i, $1 \le i \le n$, g, G, $\mathbf{z}$, $\epsilon \in \mathbb{N}_0$
Output: all $f \in \mathcal{L}(G)$ such that $\big(f(P_1), \ldots, f(P_n)\big)$ and $\mathbf{z}$ disagree in at most ϵ places

1: **Preparation Step**: Let

$$\begin{aligned}
\lambda &= \operatorname{lcm}(k_1, \ldots, k_n), \\
r &= \max\{1, 2g-1\}, \\
a &= \left((n-\epsilon)^2 - \deg G \sum_{i=1}^{n} \frac{1}{k_i} \right) \lambda^2, \\
b &= -\big(n \deg G + (2r-1)(n-\epsilon)\big)\lambda, \\
c &= r^2 - r, \\
\sigma &= \left\lfloor \frac{-b + \sqrt{b^2 - 4ac}}{2a} \right\rfloor + 1, \\
s_i &= \frac{\lambda\sigma}{k_i}, \quad 1 \le i \le n.
\end{aligned}$$

Write $(n-\epsilon)\lambda\sigma - 1 = d'_Q \deg G + r'$ with $d'_Q \ge 0$ and $0 \le r' < \deg G$. Let H be a divisor such that $\operatorname{supp} D \cap \operatorname{supp} H = \emptyset$,

$$\deg H = \begin{cases} r' & \text{if } r' \ne 0 \text{ and } r' > 2g-2, \\ \deg G & \text{otherwise,} \end{cases}$$

and let $d_Q = d'_Q$ in the first case and $d_Q = d'_Q - 1$ otherwise.

2: **Interpolation Step**: Find a non-zero polynomial $Q = \sum_{\alpha=0}^{d_Q} q_\alpha Z^\alpha \in F[Z]$ with $q_\alpha \in \mathcal{L}(H + (d_Q - \alpha)G)$ such that (P_i, z_i) is a zero of multiplicity $\ge s_i$ for all $i = 1, \ldots, n$.

3: **Factorization Step**: Calculate $f \in \mathcal{L}(G)$ such that $(Z - f) \mid Q$.

4: **return** those f obtained in Step 3 that disagree with $\mathbf{z}$ in at most ϵ places

Algorithm 5.1: Generalized Sudan decoder

that a divisor H as in Step 1 exists, since it can be chosen as the multiple of a divisor of degree 1 with support disjoint from D. Such a divisor of degree 1 can be constructed from two places of sufficiently high consecutive degree. Two places with this property do exist by Stichtenoth (1993, Corollary V.2.10). Certainly, if G itself is a multiple of a divisor of degree 1, then H can be set to a multiple of that divisor as well. The idea behind Step 1 is to ensure the existence of the desired polynomial $Q \in F[Z]$ and that $Z - f$ is a factor of such Q when f corresponds to a codeword at distance $\le \epsilon$ to $\mathbf{z}$.

An algorithm to obtain all factors of Q as constructed in Step 2 is given in (Augot and Pecquet 2000).

In the following lemma sufficient conditions for the existence of a polynomial Q as in Step 2 of Algorithm 5.1 are given. Note that the proof is constructive.

Lemma 5.1 *Let λ, σ, s_i, $1 \le i \le n$, d_Q and H be as in Step 1 of Algorithm 5.1 and $\ell = \deg H + d_Q \deg G$. If*

$$\frac{\ell(\ell+1)}{2\deg G} > \sum_{i=1}^{n} k_i \binom{s_i+1}{2} \tag{5.1}$$

then a polynomial Q as sought in Step 2 of Algorithm 5.1 does exist.

Proof: An arbitrary polynomial Q of the form $Q = \sum_{\alpha=0}^{d_Q} q_\alpha Z^\alpha \in F[Z]$ with $q_\alpha \in \mathcal{L}(G_\alpha)$ where $G_\alpha = H + (d_Q - \alpha)G$ is examined. For an arbitrary basis of $\mathcal{L}(G_\alpha)$ write $\mathcal{B}_\alpha = \{\phi_{\alpha 1}, \ldots, \phi_{\alpha \dim G_\alpha}\}$, $\mathcal{B}_\alpha^{(i)} = \{\psi_{\alpha 1}^{(i)}, \ldots, \psi_{\alpha \dim G_\alpha}^{(i)}\}$ for an increasing zero basis of $\mathcal{L}(G_\alpha)$ with respect to P_i and $\bigcup_{\alpha=0}^{d_Q} \mathcal{B}_\alpha^{(i)} = \{\psi_1^{(i)}, \ldots, \psi_\iota^{(i)}\}$. Then there exist $q_{\alpha\beta} \in \mathbb{F}_q$ such that

$$q_\alpha = \sum_{\beta=1}^{\dim G_\alpha} q_{\alpha\beta}\phi_{\alpha\beta}$$

and $\mu_{\alpha\beta\gamma}^{(i)} \in \mathbb{F}_q$ such that

$$\phi_{\alpha\beta} = \sum_{\gamma=1}^{\iota} \mu_{\alpha\beta\gamma}^{(i)} \psi_\gamma^{(i)}, \qquad \mu_{\alpha\beta\gamma}^{(i)} = 0 \text{ for } \psi_\gamma^{(i)} \notin \mathcal{B}_\alpha^{(i)}.$$

For every P_i, the polynomial Q can now be rewritten:

$$\begin{aligned}
Q &= \sum_{\alpha=0}^{d_Q} q_\alpha Z^\alpha \\
&= \sum_{\alpha=0}^{d_Q} q_\alpha \sum_{\delta=0}^{\alpha} \binom{\alpha}{\delta} (Z - z_i)^\delta z_i^{\alpha-\delta} \\
&= \sum_{\delta=0}^{d_Q} (Z - z_i)^\delta \sum_{\alpha=\delta}^{d_Q} \binom{\alpha}{\delta} z_i^{\alpha-\delta} q_\alpha \\
&= \sum_{\delta=0}^{d_Q} (Z - z_i)^\delta \sum_{\alpha=\delta}^{d_Q} \binom{\alpha}{\delta} z_i^{\alpha-\delta} \sum_{\beta=1}^{\dim G_\alpha} q_{\alpha\beta}\phi_{\alpha\beta} \\
&= \sum_{\delta=0}^{d_Q} (Z - z_i)^\delta \sum_{\alpha=\delta}^{d_Q} \binom{\alpha}{\delta} z_i^{\alpha-\delta} \sum_{\beta=1}^{\dim G_\alpha} q_{\alpha\beta} \sum_{\gamma=1}^{\iota} \mu_{\alpha\beta\gamma}^{(i)} \psi_\gamma^{(i)} \\
&= \sum_{\gamma=1}^{\iota} \sum_{\delta=0}^{d_Q} \psi_\gamma^{(i)} (Z - z_i)^\delta \sum_{\alpha=\delta}^{d_Q} \binom{\alpha}{\delta} z_i^{\alpha-\delta} \sum_{\beta=1}^{\dim G_\alpha} \mu_{\alpha\beta\gamma}^{(i)} q_{\alpha\beta}
\end{aligned}$$

Let $f \in \mathcal{L}(G)$ such that $f(P_i) = z_i$. Observe that $v_{P_i}\big(\big(f - f(P_i)\big)^\delta\big) \ge \delta$ and therefore $v_{P_i}\big(\psi_\gamma^{(i)}\big(f - f(P_i)\big)^\delta\big) \ge v_{P_i}(\psi_\gamma^{(i)}) + \delta$. The above rewriting of Q makes it now apparent that if

$$\sum_{\alpha=\delta}^{d_Q} \binom{\alpha}{\delta} z_i^{\alpha-\delta} \sum_{\beta=1}^{\dim G_\alpha} \mu_{\alpha\beta\gamma}^{(i)} q_{\alpha\beta} = 0 \qquad \text{for } v_{P_i}(\psi_\gamma^{(i)}) + \delta < s_i, \tag{5.2}$$

then $v_{P_i}\big(Q(f)\big) \geq s_i$. Thus if (5.2) holds, (P_i, z_i) is a zero of multiplicity at least s_i of Q.

The functions $\psi_\gamma^{(i)}$ have valuation ≥ 0 with respect to every P_i since $\operatorname{supp} G_\alpha \cap \operatorname{supp} D = \emptyset$. Therefore there are at most $k_i(s_i - \delta)$ functions $\psi_\gamma^{(i)}$ for every value of δ, $0 \leq \delta \leq s_i - 1$ such that $v_{P_i}(\psi_\gamma^{(i)}) + \delta < s_i$. Summing up, there are at most $k_i\binom{s_i+1}{2}$ equations in (5.2) with respect to a place P_i, putting constraints on the coefficients $q_{\alpha\beta}$ of Q. Altogether there are at most

$$\sum_{i=1}^{n} k_i \binom{s_i + 1}{2}$$

equations.

On the other hand, each q_α is a linear combination of basis elements of $\mathcal{L}(G_\alpha)$, which is of dimension $\deg G_\alpha - g + 1 = \deg H + (d_Q - \alpha)\deg G - g + 1$ since H has been chosen such that $\deg H > 2g - 2$. The coefficients $q_{\alpha\beta}$ are the unknowns and altogether there are

$$\begin{aligned}
&\sum_{\alpha=0}^{d_Q} \deg H + (d_Q - \alpha)\deg G - g + 1 \\
&\qquad = \sum_{\alpha=0}^{\frac{\ell - \deg H}{\deg G}} \ell - \alpha \deg G - g + 1 \\
&\qquad = \left(\frac{\ell - \deg H}{\deg G} + 1\right)(\ell - g + 1) - \deg G \binom{\frac{\ell - \deg H}{\deg G} + 1}{2} \\
&\qquad = \left(\frac{\ell - \deg H}{\deg G} + 1\right)\left(\ell - g + 1 - \frac{\ell - \deg H}{2}\right) \\
&\qquad = \left(\frac{\ell + \deg G - \deg H}{\deg G}\right)\left(\frac{\ell + \deg H - (2g-2)}{2}\right) \\
&\qquad \geq \frac{\ell(\ell+1)}{2\deg G}
\end{aligned}$$

of these.

Thus if (5.1) holds, there are more unknowns than linear constraints and a non-trivial solution to the homogeneous linear system of equations is guaranteed. □

Next, sufficient conditions are given for when the constructed polynomial Q has a factor of the type $Z - f$, $f \in \mathcal{L}(G)$.

Lemma 5.2 *Let λ, σ, s_i, $1 \leq i \leq n$, d_Q, H and Q be as in Algorithm 5.1. Further, let $f \in \mathcal{L}(G)$ such that $f(P_i) \neq z_i$ for at most ϵ instances of i, $1 \leq i \leq n$. If*

$$(n - \epsilon)\lambda\sigma > \deg H + d_Q \deg G, \tag{5.3}$$

then $(Z - f) \mid Q$.

Proof: Let I be the set of indices such that $f(P_i) = z_i$, i.e. $|I| \geq (n-\epsilon)$. As the functions f^α are elements of $\mathcal{L}(\alpha G)$, $Q(f) \in \mathcal{L}(H + d_Q G)$ by the choice of coefficients of Q. But since $v_{P_i}\big(Q(f)\big) \geq s_i$ for $i \in I$ by the definition of multiplicities, $Q(f)$ is also an element of the smaller space $\mathcal{L}(H + d_Q G - \sum_{i\in I} s_i P_i)$. Now if (5.3) holds, then

$$\deg\left(\sum_{i\in I} s_i P_i\right) = \sum_{i\in I} s_i k_i = |I|\sigma\lambda \geq (n-\epsilon)\sigma\lambda > \deg(H + d_Q G).$$

But $\mathcal{L}(A) = \{0\}$ for any divisor A such that $\deg A < 0$, i.e. $\mathcal{L}(H + d_Q G - \sum_{i\in I} s_i P_i) = \{0\}$. Therefore $Q(f) = 0$, implying $(Z - f) \,|\, Q$. □

Suppose that f and g are functions corresponding to two codewords at the same distance $\leq \epsilon$ to $\mathbf{z}$ and note that if the numbers s_i were not chosen to level out the degrees of the places P_i ($s_i k_i = s_j k_j$ for all i and j), then it might happen that the function g does not fulfill $\sum_{i\in I} s_i k_i > \deg H + d_Q \deg G$ as in the proof of Lemma 5.2, whereas f does. That shows that the multiplicities s_i are important to ensure that both $Z - f$ and $Z - g$ are factors of Q.

The following lemma gives sufficient conditions such that the conditions of Lemma 5.1 and Lemma 5.2 are fulfilled.

Lemma 5.3 *Again let* $\ell = \deg H + d_Q \deg G$. *If* ϵ *fulfills*

$$\epsilon < n - \sqrt{\deg G \sum_{i=1}^{n} \frac{1}{k_i}}, \tag{5.4}$$

then for the choices made in Step 1 of Algorithm 5.1, inequalities (5.1) and (5.3) both hold.

Proof: Note that, in Step 1 of Algorithm 5.1, the divisor H and the upper bound on the degree d_Q of Q have been chosen such that

$$(n-\epsilon)\lambda\sigma > \deg H + d_Q \deg G = \ell \geq (n-\epsilon)\lambda\sigma - r. \tag{5.5}$$

Therefore (5.3) holds.

To show (5.1), note that by (5.4) the number a in Step 1 is positive and that σ is chosen as the closest integer greater than the largest root of a second order degree polynomial. Therefore

$$\left((n-\epsilon)^2 - \deg G \sum_{i=1}^{n} \frac{1}{k_i}\right)\lambda^2\sigma^2 - \big(n \deg G + (2r-1)(n-\epsilon)\big)\lambda\sigma + (r^2 - r) > 0,$$

which implies

$$\frac{(n-\epsilon)^2\lambda^2\sigma^2 - (2r-1)(n-\epsilon)\lambda\sigma + (r^2-r)}{2\deg G} > \frac{\lambda^2\sigma^2}{2}\sum_{i=1}^{n}\frac{1}{k_i} + n\frac{\lambda\sigma}{2} = \sum_{i=1}^{n} k_i \binom{s_i+1}{2}.$$

On the other hand, by (5.5)

$$\frac{\ell(\ell+1)}{2\deg G} \geq \frac{\big((n-\epsilon)\lambda\sigma - r\big)\big((n-\epsilon)\lambda\sigma - r + 1\big)}{2\deg G} = \frac{(n-\epsilon)^2\lambda^2\sigma^2 - (2r-1)(n-\epsilon)\lambda\sigma + (r^2-r)}{2\deg G}.$$

Therefore, condition (5.1) is also fulfilled. □

The results of the previous lemmas can now be gathered in the main theorem of this paper.

Theorem 5.1 *If*

$$\epsilon \leq \left\lceil n - \sqrt{\deg G \sum_{i=1}^{n} \frac{1}{k_i}} \right\rceil - 1, \tag{5.6}$$

then Algorithm 5.1 recovers all functions corresponding to codewords at distance $\leq \epsilon$ *to* $\mathbf{z}$.

Certainly, d_Q, H and σ have been designed to fulfill (5.1) and (5.3). The number σ has been chosen as the smallest positive integer that fulfills (5.1). This will result in as few unknowns and equations in the linear system as possible. The numbers d_Q and $\deg H$ (and thus ℓ) on the other hand have been chosen as large as possible while still satisfying (5.3). This will allow the correction of ϵ errors (under condition (5.6)) as well as making sure that the linear system (5.2) consists of more unknowns than equations.

5.4 Representation of Functions with Respect to a Local Parameter

The method in the proof of Proposition 5.1, for finding an increasing zero basis of $\mathcal{L}(A)$ with respect to a place P_i of degree k_i given an arbitrary basis of the space, depends on whether it is possible to calculate the unique representation of a non-zero function of $\mathcal{L}(A)$ with respect to a local parameter of P_i. In this section, the procedure given by Høholdt and Nielsen (1999, Section 5) for the Hermitian function field is generalized. Suppose $F = \mathbb{F}_q(x,y)$ is defined by the equation $p(x,y) = 0$, with $p \in \mathbb{F}_q[X,Y]$ irreducible. Note that there are k_i places of degree 1 above P_i in the constant field extension $\mathbb{F}_{q^{k_i}}F$ of F. These correspond to k_i points on the curve defined by p over $\mathbb{F}_{q^{k_i}}$. Our generalization consists of performing the procedure by Høholdt and Nielsen (1999, Section 5) once on both numerator and denominator of the given function with respect to some of the places in the extension field. The result will be defined again over F. These calculations can be performed once and for all and therefore do not increase the complexity of the decoding Algorithm 5.1.

In the following, the terminology pertaining to a polynomial ring in the context of a function field is used. A function $g(x,y) \in F$ where $g \in \mathbb{F}_q[X,Y]$ will also be called a polynomial. Further, $g(x,y)$ is called irreducible, univariate

or linear if g is. The polynomial $g(x, y)$ is said to factor over a field that $\mathbb{F}_q$ is contained in if g does, and $h(x, y) \in F$, $h \in \mathbb{F}_q[X, Y]$ is a factor of $g(x, y)$ if h is one of g.

The next result will indicate under which conditions a method as described above can be employed.

Proposition 5.2 *Let $P'_1, \ldots, P'_{k_i}$ be the places above $P_i \neq P_\infty$ in the constant field extension $\mathbb{F}_{q^{k_i}}F$ of F. Further let P_i lie above the place Q_{t_x} in the rational function field $\mathbb{F}_q(x)$ corresponding to the irreducible polynomial $t_x \in \mathbb{F}_q(x)$. The polynomial t_x factors into linear polynomials $x - \nu_1, \ldots, x - \nu_{\deg t_x}$ over $\mathbb{F}_{q^{k_i}}$, and if t_x is a local parameter of P_i, then each factor is a local parameter of some P'_α.*

Proof: Recall, that $\deg Q_{t_x} = \deg t_x$ divides $\deg P_i = k_i$. Therefore, t_x factors into linear polynomials $x - \nu_1, \ldots, x - \nu_{\deg t_x}$ over $\mathbb{F}_{q^{k_i}}$. In the constant field extension $\mathbb{F}_{q^{k_i}}(x)$ of $\mathbb{F}_q(x)$ there are $\deg t_x$ places, denoted $Q'_1, \ldots, Q'_{\deg t_x}$, above Q_{t_x}, namely the ones with local parameters corresponding to the linear factors of t_x. For each $\beta = 1, \ldots, \deg t_x$ there exists α such that $P'_\alpha \mid Q'_\beta$. If t_x is not just the local parameter of Q_{t_x}, but also of P_i, then it is one of P'_α as well since the constant field extension is unramified. As $v_{P'_\alpha}(x - x_\beta) > 0$ and $v_{P'_\alpha}(x - x_\gamma) = 0$ for $\beta \neq \gamma = 1, \ldots, \deg t_x$, therefore $v_{P'_\alpha}(x - x_\beta) = v_{P'_\alpha}(t_x) = 1$ and $x - x_\beta$ is a local parameter of P'_α. □

First a remark about the above proposition. The function t_x is not only a local parameter of Q_{t_x}, but also of P_i if and only if $P_i \mid Q_{t_x}$ is unramified. This is for example the case when there are $[F : \mathbb{F}_q(x)] = \deg_Y p$ places of F above Q_{t_x} or when P_i is the only place above Q_{t_x} with relative degree $\deg_Y p$.

In the following, it is assumed that t_x of Proposition 5.2 is a local parameter of P_i. Where one of these assumptions does not hold (for example for an infinite place P_∞), the corresponding place of the function field of another one of the affine curves covering the corresponding projective curve can be considered.

Suppose the representation of an arbitrary function $f \in F$ with respect to t_x is desired. The function can be written $f = g/h$ where $g, h \in F$ are polynomials, and finding a representation means dividing g and h by t_x, keeping track of the number of divisions and stopping when units of $\mathcal{O}_{P_i}$ are obtained. Proposition 5.2 and the fact that $v_{P_i}(f) = v_{P'_\alpha}(f)$, since a constant field extension is unramified, imply that dividing g and h by t_x until units of $\mathcal{O}_{P_i}$ are obtained is the same as dividing by each linear factor of t_x until units with respect to a place P'_α having that factor as local parameter are obtained.

Consider now the case where the polynomial $g(x, y) \in \mathbb{F}_{q^{k_i}}F$, $g \in \mathbb{F}_{q^{k_i}}[X, Y]$ is to be divided by the factor $x - x_\alpha$ of t_x. Let P'_α be a place above P_i corresponding to the point (x_α, y_α) on the curve over $\mathbb{F}_{q^{k_i}}$ defined by p. Note that any such polynomial $g(x, y)$ has a representation

$$g(x, y) = \sum_{\gamma, \delta \in \mathbb{N}_0} g_{\gamma\delta}(x - x_\alpha)^\gamma (y - y_\alpha)^\delta$$

$$= g_{00} + (y - y_\alpha) \sum_{\delta \in \mathbb{N}} g_{0\delta} (y - y_\alpha)^{\delta - 1} + (x - x_\alpha) \sum_{\substack{\gamma \in \mathbb{N} \\ \delta \in \mathbb{N}_0}} g_{\gamma\delta} (x - x_\alpha)^{\gamma - 1} (y - y_\alpha)^{\delta} \tag{5.7}$$

with $g_{\gamma\delta} \in \mathbb{F}_{q^{k_i}}$. This is because g has a representation $g = \sum_{\gamma\delta \in \mathbb{N}_0} g_{\gamma,\delta} (X - x_\alpha)^\gamma (Y - y_\alpha)^\delta$ with $g_{\gamma\delta} \in \mathbb{F}_{q^{k_i}}$ by choosing $g_{00} = g(x_\alpha, y_\alpha)$ and observing that $g - g_{00}$ is contained in the maximal ideal generated by $X - x_\alpha$ and $Y - y_\alpha$.

First, such a representation of $p(x, y)$

$$p(x, y) = (y - y_\alpha) \sum_{\delta \in \mathbb{N}} p_{0\delta} (y - y_\alpha)^{\delta - 1} + (x - x_\alpha) \sum_{\substack{\gamma \in \mathbb{N} \\ \delta \in \mathbb{N}_0}} p_{\gamma\delta} (x - x_\alpha)^{\gamma - 1} (y - y_\alpha)^{\delta}$$

is considered. The term $p_{00} = 0$ since the point (x_α, y_α) lies on the curve defined by p. None of the two terms in the above expression are trivial since p is irreducible over $\mathbb{F}_q$ and therefore

$$\frac{y - y_\alpha}{x - x_\alpha} = - \frac{\sum_{\gamma \in \mathbb{N}, \delta \in \mathbb{N}_0} p_{\gamma\delta} (x - x_\alpha)^{\gamma - 1} (y - y_\alpha)^{\delta}}{\sum_{\delta \in \mathbb{N}} p_{0\delta} (y - y_\alpha)^{\delta - 1}}. \tag{5.8}$$

In the following, it is assumed that the denominator on the right hand side $w = \sum_{\delta \in \mathbb{N}} p_{0\delta} (y - y_\alpha)^{\delta - 1}$, is a unit of $\mathcal{O}_{P'_\alpha}$. This is the case if and only if $p_{01} \neq 0$.

A polynomial $g(x, y)$ as in (5.7) is contained in $P_i \subseteq P'_\alpha$ if and only if $g_{00} = 0$, and can therefore be divided by $x - x_\alpha$ by simple reduction or with the help of (5.8). After this division, the result g' is not necessarily a polynomial anymore and a new representation is needed. It can be written as

$$\begin{aligned} g' &= \sum_{\gamma, \delta \in \mathbb{N}_0} g'_{\gamma\delta} (x - x_\alpha)^\gamma (y - y_\alpha)^\delta \\ &= g'_{00} + (y - y_\alpha) \sum_{\delta \in \mathbb{N}} g'_{0\delta} (y - y_\alpha)^{\delta - 1} \\ &\quad + (x - x_\alpha) \sum_{\substack{\gamma \in \mathbb{N} \\ \delta \in \mathbb{N}_0}} g'_{\gamma\delta} (x - x_\alpha)^{\gamma - 1} (y - y_\alpha)^{\delta} \end{aligned} \tag{5.9}$$

where $g'_{\gamma\delta} = \sum_{\iota \in \mathbb{Z}} g'_{\gamma\delta\iota} w^\iota$, $g'_{\gamma\delta\iota} \in \mathbb{F}_{q^{k_i}}$.

Again, $g' \in P'_\alpha$ if and only if $g'_{00}(P'_\alpha) = 0$ and then division of the terms involving $x - x_\alpha$ and $y - y_\alpha$ can be done by simple reduction or with the help of (5.8). For g'_{00} an extra effort needs to be made. Let $\kappa = \min\{\iota \in \mathbb{Z} \mid g'_{00\iota} \neq 0\}$. Then $g'_{00} = w^\kappa \sum_{\iota \in \mathbb{N}_0} g'_{00\iota+\kappa} w^\iota$. The function $\sum_{\iota \in \mathbb{N}_0} g'_{00\iota+\kappa} w^\iota$ is a univariate polynomial in y, and since $g'_{00}(P'_\alpha) = 0$ it must have $y - y_i$ as a factor. Rewrite

$$\frac{g'_{00}}{x - x_\alpha} = w^\kappa \frac{y - y_\alpha}{x - x_\alpha} e = -w^{\kappa - 1} \left(\sum_{\substack{\gamma \in \mathbb{N} \\ \delta \in \mathbb{N}_0}} p_{\gamma\delta} (x - x_\alpha)^{\gamma - 1} (y - y_\alpha)^{\delta} \right) e \tag{5.10}$$

for some univariate polynomial $e \in \mathbb{F}_{q^{k_i}} F$ in y.

Input: $f = g/h \in F$, g, h polynomials, local parameter t_x of the place P_i and points (x_α, y_α), $1 \leq \alpha \leq \deg t_x$ on the curve with distinct first coordinates such that $x - x_\alpha$ is a factor of t_x corresponding to places P'_α above P_i
Output: unit $u \in \mathcal{O}_{P_i}$ and $m \in \mathbb{Z}$ such that $f = ut_x^m$

1: Initialize $g^{(0)} = g$, $h^{(0)} = h$, $m_1 = 0$ and $m_2 = 0$;
2: **for** $\alpha = 1$ to $\deg t_x$ **do**
3: Find representations for $g^{(0)}$ and $h^{(0)}$ as in (5.7).
4: **while** $g_{00}^{(m_1)}(P'_\alpha) \neq 0$ **do**
5: Calculate

$$g^{(m_1+1)} = \frac{g^{(m_1)}}{x - x_\alpha}$$

with the help of (5.8) and (5.10) and obtain a representation of $g^{(m_1+1)}$ as in (5.9).
6: Increase m_1 by 1.
7: **end while**
8: **while** $h_{00}^{(m_2)}(P'_\alpha) \neq 0$ **do**
9: Calculate

$$h^{(m_2+1)} = \frac{h^{(m_2)}}{x - x_\alpha}$$

with the help of (5.8) and (5.10) and obtain a representation of $h^{(m_2+1)}$ as in (5.9).
10: Increase m_2 by 1.
11: **end while**
12: Let $g^{(m_1)}/h^{(m_2)} = g^{(0)}/h^{(0)}$ such that $g^{(0)}, h^{(0)}$ are polynomials.
13: **end for**
14: **return** $g^{(m_1)}/h^{(m_2)}$ and $m_1 - m_2$.

Algorithm 5.2: Representation with respect to a local parameter

The method for obtaining a representation of an arbitrary function of F with respect to the local parameter t_x of P_i is stated in Algorithm 5.2. Note that it is conditioned on the fact that the places concerned lie unramified above places of one of the rational function fields contained in F and that w is a unit of $\mathcal{O}_{P'_\alpha}$. This depends on the representation (5.7) of the defining polynomial $p(x, y)$.

Due to symmetry, the above can be restated analogously with respect to a similar local parameter t_y of the place Q_{t_y} in $\mathbb{F}_q(y)$ below P_i.

5.5 Example

Consider the elliptic function field $F = \mathbb{F}_4(x, y)$ defined by the equation $y^2 + y = x^3 + x$. Its Zeta function is

$$Z(t) = \frac{4t^2 + 1}{(1 - t)(1 - 4t)} = \exp\Big(\sum_{i \geq 1} \frac{N_i}{i} t^i\Big),$$

where N_i denotes the number of rational places of $\mathbb{F}_{4^i}F$. Expanding gives

$$t \cdot \frac{d}{dt} \log Z(t) = \sum_{i \geq 1} N_i t^i = 5t + 25t^2 + 65t^3 + 225t^4 + \cdots$$

which indicates that $B_1 = 5$, $B_2 = 10$, $B_3 = 20$, $B_4 = 50$ are the numbers of places of F of degrees 1 up to 4. Using all 5 and 10 places of degree 1 and 2 plus 19 of degree 3 for evaluation gives a generalized geometric Goppa code $\mathcal{C}$. Concatenating subsequently with linear $[1,1,1]$, $[3,2,2]$, $[5,3,3]$ codes over $\mathbb{F}_4$ respectively, results in a code over $\mathbb{F}_q$ with parameters $[130, k, d \geq 82 - k]$, $1 \leq k \leq 81$ (Ding, Niederreiter, and Xing 2000). For $k = 29$, a new $[130, 29, d \geq 53]$ code over $\mathbb{F}_4$ is obtained. In this section, some aspects of the construction and the decoding of the code $\mathcal{C}$ are illustrated explicitly.

To start with, a basis of the space $\mathcal{L}\big(k(P^{(4)} - P^{(3)})\big)$ needs to be constructed, where $P^{(3)}$ and $P^{(4)}$ are places of degree 3 and 4 respectively such that $P^{(3)}$ is not used for evaluation. Suppose $f \in F$ such that $(f) = P^{(3)} + P_\infty - P^{(4)}$, then $f^k \in \mathcal{L}\big(k(P^{(4)} - P^{(3)})\big)$ and

$$\mathcal{L}\big(k(P^{(4)} - P^{(3)})\big) = f^k \mathcal{L}(kP_\infty)$$

by the following: The two spaces have equal dimension and thus by proving the inclusion $\supseteq$, the equality is established: $g \in f^k \mathcal{L}(kP_\infty) \Rightarrow g = f^k \cdot h$, where $h \in \mathcal{L}(kP_\infty) \Rightarrow g = f^k \cdot h$ where $(h) \geq -kP_\infty \Rightarrow (g) \geq k(P^{(3)} - P^{(4)}) \Rightarrow g \in \mathcal{L}\big(k(P^{(4)} - P^{(3)})\big)$. By (Stichtenoth 1993, VI.4.1. (h)) $\{x^i y^j \mid 0 \leq i,\ j = 0,\ 1,\ 2i + 3j \leq k\}$ is a basis of $\mathcal{L}(kP_\infty)$, and thus $B = \{f^k x^i y^j \mid 0 \leq i,\ j = 0,\ 1,\ 2i + 3j \leq k\}$ is a basis of $\mathcal{L}\big(k(P^{(4)} - P^{(3)})\big)$.

It remains to construct f. Let

$$f = \frac{y + \mu}{y + \mu(1 + x + x^2)}$$

where μ is a primitive element of $\mathbb{F}_4$. Substituting μ and $\mu(1 + x + x^2)$ for y in the equation of the curve, irreducible polynomials of $\mathbb{F}_4[x]$ of degree 3 and 4 respectively are obtained. Thus $(y + \mu)_0 = P^{(3)}$, $\big(y + \mu(1 + x + x^2)\big)_0 = P^{(4)}$ where $P^{(3)}$, $P^{(4)}$ are places of degree 3 and 4 respectively. Now $v_{P_\infty}(x) = -2$ and $v_{P_\infty}(y) = -3$ and therefore $(y + \mu)_\infty = 3P_\infty$, $\big(y + \mu(1 + x + x^2)\big)_\infty = 4P_\infty$ and $(f) = P^{(3)} + P_\infty - P^{(4)}$.

By the above,

$$\begin{aligned}\mathcal{L}\big(29(P^{(4)} - P^{(3)})\big) = \langle & f^{29}, f^{29}x, f^{29}x^2, \ldots, f^{29}x^{14}, \\ & f^{29}y, f^{29}xy, \ldots, f^{29}x^{12}y, f^{29}x^{13}y \rangle .\end{aligned}$$

Evaluation of functions in $\mathcal{L}\big(29(P^{(4)} - P^{(3)})\big)$ in the places of degree 1, 2 and 3 apart from the one infinite place P_∞ is straight forward. The variables x, y in the given function are substituted by the coordinates of fixed points on the curve over $\mathbb{F}_4, \mathbb{F}_{4^2}$ and $\mathbb{F}_{4^3}$ respectively that correspond to places above the place considered in the various constant field extensions. The place P_∞ however corresponds the point $(0 : 1 : 0)$ on the corresponding projective curve

$y^2z + yz^2 = x^3 + xz^2$. Evaluation of elements of $\mathcal{L}\big(29(P^{(4)} - P^{(3)})\big)$ at P_∞ is done by evaluation of the homogenized function at $(0:1:0)$. Homogenizing f gives

$$\begin{aligned} f^* &= \frac{yz + \mu z^2}{yz + \mu(z^2 + xz + x^2)} \\ &= \frac{xy + \mu xz}{xy + \mu(y^2 + yz + x^2)}, \end{aligned}$$

using that $x^2 + z^2 = (y^2z + yz^2)/x$. Naturally, the homogenized functions are linear combinations of the homogenized basis $B^* = \{f^{*29}, \ldots, f^{*29}x^{13}y/z^{14}\}$ of $\mathcal{L}\big(29(P^{(4)} - P^{(3)})\big)$.

To create rows of the generator matrix of the outer code B^*, is evaluated in $(0:1:0)$ and B in the 33 places of degree 1,2 and 3.

The following inner codes can be used for the concatenation: the trivial $[1, 1, 1]$ code over $\mathbb{F}_4$ for coordinates corresponding to places of degree 1, the $[3, 2, 2]$ code over $\mathbb{F}_4$ with generator and parity check matrices

$$G_2 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}, \quad H_2 = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$$

for those corresponding to places of degree 2, and the doubly extended $[5, 3, 3]$ Reed-Solomon code over $\mathbb{F}_4$ (MacWilliams and Sloane 1977, Chapter 11, §5) with generator and parity check matrices

$$G_3 = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & \mu & \mu^2 & 0 \\ 0 & 1 & \mu^2 & \mu & 1 \end{pmatrix}, \; H_3 = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & \mu & \mu^2 & 1 \end{pmatrix}.$$

These can easily be decoded by syndrome decoding.

As far as the outer decoder Algorithm 5.1, is concerned a few remarks might be useful. Since $G = 29(P^{(4)} - P^{(3)})$ is a multiple of the divisor $P^{(4)} - P^{(3)}$ of degree 1, H in the preparation step can be chosen as $H = r'(P^{(4)} - P^{(3)})$ in the first case and $H = G$ in the other.

For the purpose of solving the linear system of equation (5.2) in the interpolation step, increasing zero bases of $\mathcal{L}\big(H + (d_Q + j)G\big)$, $1 \le j \le d_Q$ have to be computed. The divisor $H + (d_Q + j)G$ is a multiple of $P^{(4)} - P^{(3)}$ as well, and bases of the spaces can be obtained in a similar way as the one of $\mathcal{L}\big(29(P^{(4)} - P^{(3)})\big)$ has been constructed. Suppose now that such bases are available and increasing zero bases of each space with respect to each of the 34 places have to be calculated.

Note that apart from the infinite place, all places of degree 1, 2 and 3 lie unramified above a place of one of the rational function fields contained in F. For the infinite place, the point $(0, 0)$ on the affine curve $x^3 + z^2 + xz^2 + z$ has to be considered. This is one of the three affine curves that cover the corresponding projective curve. The polynomial $p(x, y)$ that defines the function field can be represented as in (5.7)

$$p(x,y) = \begin{cases} (x + x_\alpha)^3 + x_\alpha(x + x_\alpha)^2 + x_\alpha^2(x + x_\alpha) & \\ \qquad + (y + y_\alpha)^2 + (y + y_\alpha) & x_\alpha \neq 0 \\ x^3 + x + (y + y_\alpha)^2 + (y + y_\alpha) & x_\alpha = 0 \end{cases}$$

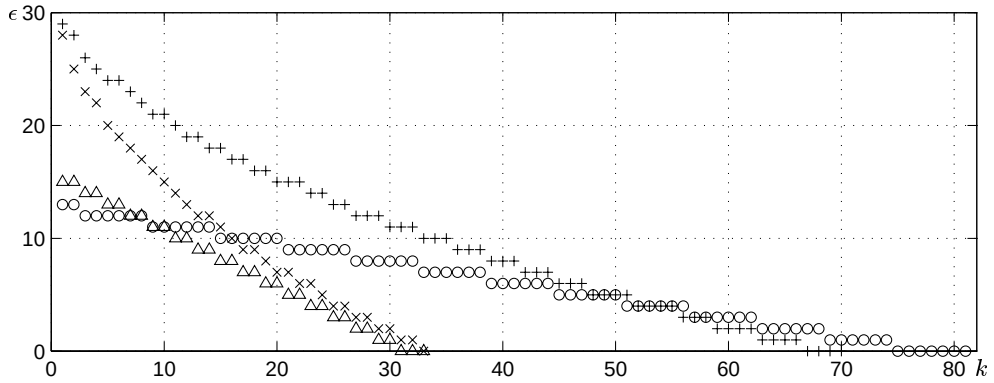


Figure 5.1: Error-correcting capacities: (5.6) and (4.4) for the outer, length 34 code of this section as a function in $k = \deg G$ illustrated by '+' and 'o' respectively, (5.6) and (4.4) for a conventional length 34 geometric Goppa code over a function field with $g = 1$ as a function in $k = \deg G$ illustrated by '×' and '△' respectively.

for an arbitrary point (x_α, y_α) lying on the curve over any extension field of $\mathbb{F}_q$ and therefore division by $x + x_\alpha$ or $y + y_\alpha$ can be performed with the help of (5.8).

Consider a place of degree 3. The place $P^{(3)}$ above the places Q_{x^3+x+1} and $Q_{y+\mu}$ of the rational function fields $\mathbb{F}_4(x)$ and $\mathbb{F}_4(y)$ respectively has already been used to construct the divisor G. There is also another place $\bar{P}^{(3)}$ of F above the place Q_{x^3+x+1}, namely the one that also lies above the place $Q_{y+\mu^2}$ of $\mathbb{F}_4(y)$. Since there are $[F : \mathbb{F}_4(x)] = 2$ distinct places above Q_{x^3+x+1}, x^3+x+1 is a local parameter of $\bar{P}^{(3)}$. The polynomial factors into three linear polynomials $x - \nu_1$, $x - \nu_2$, $x - \nu_3$ over $\mathbb{F}_{4^3}$ corresponding to three places above $\bar{P}^{(3)}$ in $\mathbb{F}_{q^3}F$, and the points on the corresponding curve over $\mathbb{F}_{4^3}$ are $(\nu_1, \mu^2), (\nu_2, \mu^2), (\nu_3, \mu^2)$ as needed in the input of Algorithm 5.2. Actually, $y + \mu^2$ is also a local parameter of $\bar{P}^{(3)}$, since it is the only place of F above $Q_{y+\mu^2}$ and the relative degree of $\bar{P}^{(3)}$ with respect to $Q_{y+\mu^2}$ is $[F : \mathbb{F}_4(y)] = 3$. So, Algorithm 5.2 can be performed with the single point (ν_1, μ^2) as input, dividing numerator and denominator by $t_y = y + \mu^2$.

The error-correcting capacity (5.6) of Algorithm 5.1 for the outer code of length 34 and dimension $1 \leq k = \deg G \leq 81$ discussed in this section is illustrated in Figure 5.1 and compared to (4.4) as well as the corresponding error-correcting capacities for a conventional geometric Goppa code of length 34 with $g = 1$, $1 \leq k = \deg G \leq 33$. Such a code can for example be constructed by considering the constant field extension $\mathbb{F}_{4^6}F$ and 34 of its rational places for evaluation. This is somewhat surprising as the generalized geometric Goppa code and the conventional one over a bigger base field seem to be identical at first sight. Of course they are not as one is a code over $\mathbb{F}_4$ and the other over $\mathbb{F}_{4^6}$.

Several parts of the calculations for this example have been implemented in the computer algebra system SINGULAR (Greuel et al. 1998).

5.6 Conclusions

A polynomial time algorithm to decode generalized geometric Goppa codes has been developed. This algorithm is a generalization of Sudan's improved algorithm (Guruswami and Sudan 1999). The analysis of the error-correcting capacity of the algorithm, Theorem 5.1, reveals a considerable improvement for some dimensions compared to the method in (Heydtmann 2000), see Figure 5.1. This algorithm uses the structure of the code as a subcode over $\mathbb{F}_q$ of a conventional generalized geometric Goppa code over $\mathbb{F}_{q^\ell}$ considered as a space over $\mathbb{F}_q$ as well. Also, the bound in Theorem 5.1 on the error-correcting capacity simplifies to $\epsilon < n - \sqrt{\deg G n}$ in the case of conventional geometric Goppa codes, which is precisely the bound obtained in (Guruswami and Sudan 1999). It makes apparent that the usage of places of higher degree make it possible to correct more errors. Then again, working with higher degree places is also paid for by having to represent the codeword entries that lie in extension fields of the base field.

Further, the concept of increasing zero bases of a function space has been generalized to places of higher degree in Proposition 5.1. In order to generate such bases, it is necessary to be able to represent functions with respect to a local parameter. A method with for function fields fulfilling certain properties this purpose has been presented.

Acknowledgment

The author wishes to thank Iwan M. Duursma for his help in constructing the example, Tom Høholdt and Rasmus R. Nielsen for fruitful discussions.

References

Augot, D. and L. Pecquet (2000). A Hensel Lifting to Replace Factorization in List-Decoding of Algebraic-Geometric and Reed-Solomon Codes. *IEEE Transactions on Information Theory 46*(7), 2605–2614.

Brouwer, A. E. (1998). Bounds on the Size of Linear Codes. In V. S. Pless and W. Huffman (Eds.), *Handbook of Coding Theory*, pp. 295–461. North Holland. Updated tables are available at `http://www.win.tue.nl/~aeb/voorlincod.html`.

Ding, C., H. Niederreiter, and C. Xing (2000). Some New Codes from Algebraic Curves. *IEEE Transactions on Information Theory 46*(7), 2638–2642.

Ericson, T. (1988). A Simple Analysis of the Blokh-Zyablov Decoding Algorithm. In T. Beth and M. Clausen (Eds.), *Applicable Algebra, Error-Correcting Codes, Combinatorics and Computer Algebra, 4th International Conference, Proceedings*, Volume 307 of *Lecture Notes in Computer Science*, pp. 43–57. Springer.

Garcia, A. and H. Stichtenoth (1995). A Tower of Artin-Schreier Extensions of

Function Fields Attaining the Drinfeld-Vladut Bound. *Inventiones mathematicae 121*, 211–222.

Greuel, G.-M., G. Pfister, and H. Schönemann (1998). Singular Version 1.2 User Manual. In *Reports on Computer Algebra*, Number 21. Centre for Computer Algebra, University of Kaiserslautern.
`http://www.singular.uni-kl.de`.

Guruswami, V. and M. Sudan (1999). Improved Decoding of Reed-Solomon and Algebraic-Geometric Codes beyond the Error-Correction Bound. *IEEE Transactions on Information Theory 45*(6), 1757–1767.

Heydtmann, A. E. (2000). Generalized Geometric Goppa Codes. Submitted to *Communications in Algebra*.

Høholdt, T. and R. R. Nielsen (1999). Decoding Hermitian Codes with Sudan's Algorithm. In *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, Lecture Notes in Computer Science, pp. 260–270. Springer.

MacWilliams, F. J. and N. J. A. Sloane (1977). *The Theory of Error-Correcting Codes*. North-Holland Mathematical Library.

Nielsen, R. R. (2000). Decoding Concatenated Codes with Sudan's Algorithm. Submitted to *IEEE Transactions on Information Theory*.

Nielsen, R. R. (2001). Decoding Generalized Algebraic-Geometry Codes. manuscript.

Stichtenoth, H. (1993). *Algebraic Function Fields and Codes*. Springer.

Tsfasman, M. A., S. G. Vladut, and T. Zink (1982). Modular Curves, Shimura Curves and Goppa Codes better than the Varshamov-Gilbert Bound. *Mathematische Nachrichten 109*, 21–28.

Xing, C., H. Niederreiter, and K. Y. Lam (1999). A Generalization of Algebraic-Geometry Codes. *IEEE Transactions on Information Theory 45*(7), 2498–2501.

BIBLIOGRAPHY